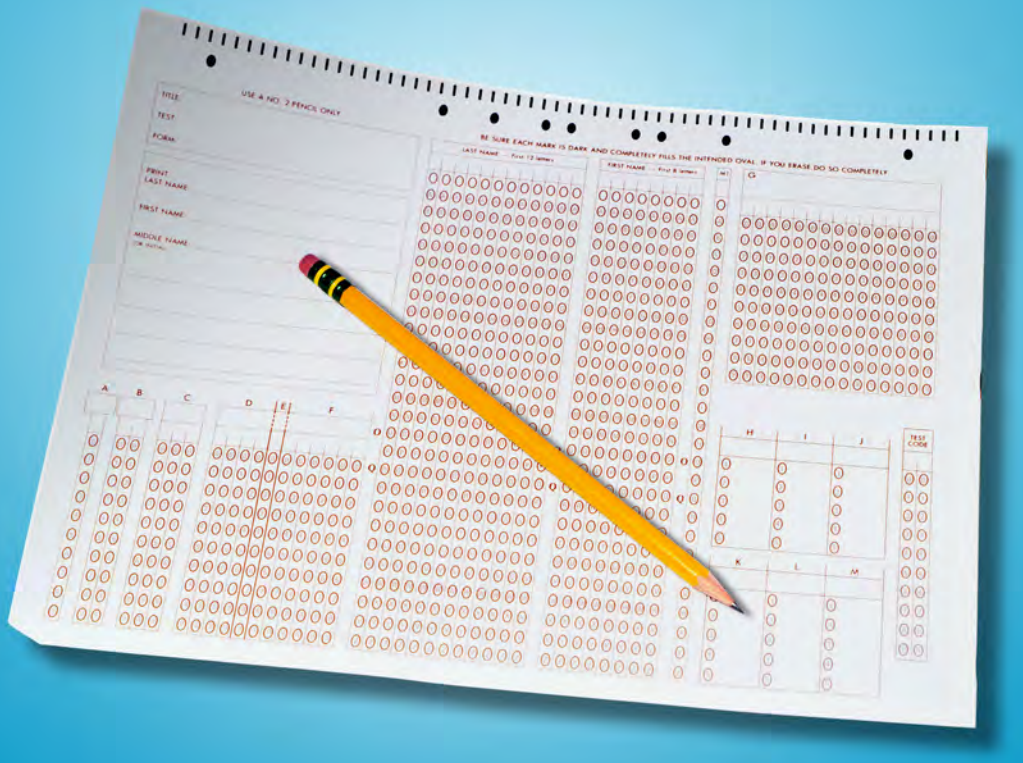
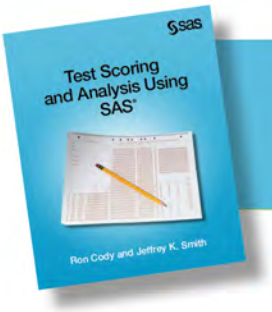
Test Scoring and Analysis Using SAS®



Ron Cody and Jeffrey K. Smith



Contents

List of Programs vii
About This Book xi
About These Authors xv
Acknowledgments xvii
Chapter 1: What This Book Is About 1
Introduction 1
An Overview of Item Analysis and Test Reliability 1
A Brief Introduction to SAS 2
Chapter 2: Reading Test Data and Scoring a Test 5
Introduction 5
Reading Data from a Text File and Scoring a Test 6
Explanation of Program 2.1 8
Reading Space-Delimited Data 10
Reading Comma-Delimited Data (CSV File) 12
Reading Data Directly from an Excel Workbook 14
Reading an Answer Key from a Separate File 16
Modifying the Program to Score a Test of an Arbitrary Number of Items 17
Displaying a Histogram of Test Scores 20
Matching Student Names with Student IDs 24
Creating a Fancier Roster Using PROC REPORT 27
Exporting Your Student Roster to Excel 28
Conclusion 29

Chapter 3: Computing and Displaying Answer Frequencies 31
Introduction 31
Displaying Answer Frequencies (in Tabular Form) 31
Modifying the Program to Display the Correct Answer in the Frequency Tables 33
Developing an Automated Program to Score a Test and Produce Item Frequencies 34
Displaying Answer Frequencies in Graphical Form 36
Conclusion 40
Chapter 4: Checking Your Test Data for Errors 41
Introduction 41
Detecting Invalid IDs and Answer Choices 42
Checking for ID Errors 43
Using “Fuzzy” Matching to Identify an Invalid ID 45
Checking for and Eliminating Duplicate Records 47
Conclusion 49
Chapter 5: Classical Item Analysis 51
Introduction 51
Point-Biserial Correlation Coefficient 51
Making a More Attractive Report 53
The Next Step: Restructuring the Data Set 54
Displaying the Mean Score of the Students Who Chose Each of the Multiple Choices 55
Combining the Mean Score per Answer Choice with Frequency Counts 61
Computing the Proportion Correct by Quartile 63
Combining All the Item Statistics in a Single Table 66
Interpreting the Item Statistics 72
Conclusion 73
Chapter 6: Adding Special Features to the Scoring Program 75
Introduction 75
Modifying the Scoring Program to Accept Alternate Correct Answers 76
Deleting Items and Rescoring the Test 78
Analyzing Tests with Multiple Versions (with Correspondence Information in a Text File) 80
Analyzing Tests with Multiple Versions (with Correspondence Information in an Excel File) 82

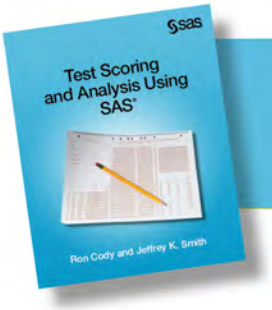
Analyzing Tests with Multiple Versions (with Correspondence Information and Student Data in an Excel File) 84
Conclusion 86
Chapter 7: Assessing Test Reliability 87
Introduction 87
Computing Split-Half Reliability 88
Computing Kuder-Richardson Formula 20 (KR-20) 92
Computing Cronbach's Alpha 93
Demonstrating the Effect of Item Discrimination on Test Reliability 94
Demonstrating the Effect of Test Length on Test Reliability 95
Conclusion 95
Chapter 8: An Introduction to Item Response Theory - PROC IRT 97
Introduction 97
IRT basics 99
Looking at Some IRT Results 100
What We Aren't Looking At! 102
Preparing the Data Set for PROC IRT 102
Running PROC IRT 104
Running Other Models 111
Classical Item Analysis on the 30-Item Physics Test 112
Conclusion 113
References 113
Chapter 9: Tips on Writing Multiple-Choice Items 115
Introduction 115
Getting Started/Organized 116
Types of Items for Achievement Tests 117
Conclusion 122
References 122
Chapter 10: Detecting Cheating on Multiple- Choice Tests 123
Introduction 123
How to Detect Cheating: Method One 123
How to Detect Cheating: Method Two 131

Searching for a Match 136
Conclusion 141
References 141

Chapter 11: A Collection of Test Scoring, Item Analysis, and Related Programs 143

Introduction 144
Scoring a Test (Reading Data from a Text File) 144
Scoring a Test (Reading Data From an Excel File) 146
Printing a Roster 148
Data Checking Program 150
Item Analysis Program 151
Program to Delete Items and Rescore the Test 154
Scoring Multiple Test Versions (Reading Test Data and Correspondence Data from Text Files) 155
Scoring Multiple Test Versions (Reading Test Data from a Text File and Correspondence Data from an Excel File) 157
Scoring Multiple Test Versions (Reading Test Data and Correspondence Data from Excel Files) 159
KR-20 Calculation 161
Program to Detect Cheating (Method One) 163
Program to Detect Cheating (Method Two) 166
Program to Search for Possible Cheating 170
Conclusion 173

Index 175



Chapter 5: Classical Item Analysis

Introduction 51
Point-Biserial Correlation Coefficient 51
Making a More Attractive Report 53
The Next Step: Restructuring the Data Set 54
Displaying the Mean Score of the Students Who Chose Each of the Multiple Choices 55
Combining the Mean Score per Answer Choice with Frequency Counts 61
Computing the Proportion Correct by Quartile 63
Combining All the Item Statistics in a Single Table 66
Interpreting the Item Statistics 72
Conclusion 73

Introduction

This chapter investigates some traditional methods of determining how well items are performing on a multiple-choice (or true/false) test. A later chapter covers advances in item response theory.

Point-Biserial Correlation Coefficient

One of the most popular methods for determining how well an item is performing on a test is called the *point-biserial correlation coefficient*. Computationally, it is equivalent to a Pearson correlation between an item response (correct=1, incorrect=0) and the test score for each student. The simplest way for SAS to produce point-biserial coefficients is by using PROC CORR. Later in this chapter, you will see a program that computes this value in a DATA step and displays it in a more compact form than PROC CORR. The following program produces correlations for the first 10 items in the statistics test described in Chapter 2.

Program 5.1: Computing Correlations Between Item Scores and Raw Scores

```
title "Computing Point-Biserial Correlations";
proc corr data=score nosimple;
   var Score1-Score10;
   with Raw;
run;
```

When you supply PROC CORR with a VAR statement and a WITH statement, it computes correlations between every variable listed on the VAR statement and every variable listed on the WITH statement. If you only supply a VAR statement, PROC CORR computes a correlation matrix—the correlation of every variable in the list with every other variable in the list.

Output from Program 5.1:

Computing Point-Biserial Correlations

The CORR Procedure

1 With Variables:	Raw
10 Variables:	Score1 Score2 Score3 Score4 Score5 Score6 Score7 Score8 Score9 Score10

Pearson Correlation Coefficients, N = 137
Prob > |r| under H0: Rho=0

	Score1	Score2	Score3	Score4	Score5	Score6	Score7	Score8	Score9	Score10
Raw Raw score	0.48830 <.0001	0.29773 0.0004	0.32225 0.0001	0.43235 <.0001	0.35183 <.0001	0.20284 0.0174	0.41984 <.0001	0.06532 0.4483	0.27607 0.0011	0.38473 <.0001

The top number in each box is the correlation between the item and the raw test score, referred to as a *point-biserial correlation coefficient*. The number below this is the *p*-value (significance level). How do you interpret this correlation? One definition of a "good" item is one where good students (those who did well on the test) get the item correct more often than students who do poorly on the test. This condition results in a positive point-biserial coefficient. Since the distribution of test scores is mostly continuous and the item scores are dichotomous (0 or 1), this correlation is usually not as large as one between two continuous variables. What does it tell you if the point-biserial correlation is close to 0? It means the "good" and "poor" students are doing equally well answering the item, meaning that the item is not helping to discriminate between good and poor students. What about negative coefficients? That situation usually results from several possible causes: One possibility is that there is a mistake in the answer key—good students are getting it wrong quite frequently (they are actually choosing the correct answer, but it doesn't match the answer key) and poor students are guessing their answers and getting the item right by chance. Another possibility is a poorly written item that good students are “reading into” and poor students are not. For example, there might be an item that uses absolutes such as "always" or "never" and the better students can think of a rare exception and do not choose the answer you expect. A third possibility is that the item is measuring something other than, or in addition to, what you are

interested in. For example, on a math test, you might have a word problem where the math is not all that challenging, but the language is a bit subtle. Thus, students with better verbal skills are getting the item right as opposed to those with better math skills. Sometimes an answer that you thought was incorrect might be appealing to the better students, and upon reflection, you conclude, "Yeah, that *could* be seen as correct." There are other possibilities as well, which we will explore later.

Making a More Attractive Report

Although the information in the previous output has all the values you need, it is hard to read, especially when there are a lot of items on the test. The program shown next produces a much better display of this information.

Program 5.2: Producing a More Attractive Report

```
proc corr data=score nosimple noprint
          outp=corrout;
   var Score1-Score10;
   with Raw;
run;
```

The first step is to have PROC CORR compute the correlations and place them in a SAS data set. To do this, you use an OUTP= procedure option. This places information about the selected variables, such as the correlation coefficients and the means, into the data set you specify with this option. The NOPRINT option is an instruction to PROC CORR that you do not want any printed output, just the data set. Here is a listing of data set CORROUT:

Listing of Data Set CORROUT

TYPE	_NAME_	Score1	Score2	Score3	Score4	Score5	Score6	Score7	Scor
MEAN		0.620	0.818	0.934	0.460	0.854	0.956	0.635	0.83
STD		0.487	0.388	0.249	0.500	0.354	0.205	0.483	0.375
N		137.000	137.000	137.000	137.000	137.000	137.000	137.000	137.000
CORR	Raw	0.488	0.298	0.322	0.432	0.352	0.203	0.420	0.065

Note: The last few columns were deleted to allow the table to fit better on the page.

This data set contains the mean, the standard deviation, N, and a correlation for each of the Score variables. The SAS created variable, _TYPE_, identifies which of these values you are looking at—the variable _NAME_ identifies the name of the WITH variable. In this example, you are only interested in the correlation coefficients. An easy way to subset the CORROUT data so that it only contains correlation coefficients is with a WHERE= data set option following the data set name. The following program is identical to Program 5.2 with the addition of the WHERE= data set option. Here is the modified program, followed by a listing of the CORROUT data set:

Program 5.3: Adding a WHERE= Data Set Option to Subset the SAS Data Set

```
proc corr data=score nosimple noprint
          outp=corrout(where=(_type_='CORR'));
   var Score1-Score10;
   with Raw;
run;
```

By using the WHERE= data set option, you now have only the correlation data in the output data set.

Listing of Data Set CORROUT (Created by Program 5.3)

TYPE	_NAME_	Score1	Score2	Score3	Score4	Score5	Score6	Score7	Score8
CORR	Raw	0.48830	0.29773	0.32225	0.43235	0.35183	0.20284	0.41984	0.065…

Note: The last few columns were deleted to allow the table to fit better on the page.

Program 5.3 is a good example of programming efficiently—using a WHERE= data set option when the data set is being created rather than writing a separate data set to create the subset.

The Next Step: Restructuring the Data Set

The next step in creating your report is to restructure (transpose) the data set above into one with one observation per item. Although you could use PROC TRANSPOSE to do this, an easier (at least to these authors) method is to use a DATA step, as follows:

Program 5.4: Restructuring the Correlation Data Set to Create a Report

```
data corr;
   set corrout;
   array Score[10];
   do Item=1 to 10;
      Corr = Score[Item];
      output;
   end;
   keep Item Corr;
run;
```

You read in one observation from data set CORROUT and, inside the DO loop, you write out one observation for each of the 10 correlations. The newly created data set (CORR) looks like this:

Listing of Data Set CORR

Item	Corr
1	0.48830
2	0.29773
3	0.32225
4	0.43235
5	0.35183
6	0.20284
7	0.41984
8	0.06532
9	0.27607
10	0.38473

You now have the 10 point-biserial coefficients in a more compact, easier-to-read format. In a later section, this information is combined with item frequencies in a single table.

Displaying the Mean Score of the Students Who Chose Each of the Multiple Choices

One interesting way to help you determine how well your items are performing is to compute the mean score for all of the students choosing each of the multiple-choice items.

To demonstrate this, let's start out with test data from a real test (a biostatistics test taken by students at a medical school in New Jersey—all the IDs have been replaced with random digits). A listing of the first 10 students with responses to the first eight items on the test are shown below:

Listing of the First Eight Items for 10 Students

ID	Ans1	Ans2	Ans3	Ans4	Ans5	Ans6	Ans7	Ans8	Raw	Percent
203579875	D*	C*	E*	B*	C*	B*	A*	B*	47	83.9286
116841443	D*	C*	E*	A	C*	B*	A*	B*	41	73.2143
176786926	D*	C*	B	B*	C*	B*	B	D	45	80.3571
011413555	D*	C*	E*	A	C*	B*	E	B*	39	69.6429
051502147	E	E	E*	B*	C*	B*	A*	B*	41	73.2143
069456918	D*	C*	E*	B*	C*	B*	A*	B*	46	82.1429
743222519	D*	C*	E*	B*	C*	B*	B	B*	33	58.9286
177458805	E	C*	E*	B*	C*	B*	A*	B*	42	75.0000

In this data set, the correct answer choice includes the letter (A through E) followed by an asterisk (as described in Chapter 3). You can run PROC FREQ on the answer variables like this. (Note: Frequencies for only the first four items were requested to limit the size of the output.)

Program 5.5: Using PROC FREQ to Determine Answer Frequencies

```
title "Frequencies for the First 4 Items on the Biostatistics Test";
proc freq data=score;
   tables Ans1-Ans4 / nocum;
run;
```

You now have answer frequencies with the correct answer to each item displayed with an asterisk.

Answer Frequencies for the First Four Items on the Biostatistics Test

Frequencies for the First 4 Items on the Biostatistics Test

The FREQ Procedure

Ans1	Frequency	Percent
A	1	0.73
B	2	1.46
D*	85	62.04
E	49	35.77

Ans2	Frequency	Percent
A	2	1.47
B	7	5.15
C*	112	82.35
D	1	0.74
E	14	10.29
Frequency Missing = 1		

Ans3	Frequency	Percent
B	7	5.11
C	2	1.46
E*	128	93.43

Ans4	Frequency	Percent
A	23	16.91
B*	63	46.32
C	24	17.65
D	17	12.50
E	9	6.62
Frequency Missing = 1		

The next step is to compute the mean score for all students who chose each of the multiple-choice answers. To do this, you must first restructure the SCORE data set so that you have one observation per student per item. The program to accomplish this restructuring is displayed next:

Program 5.6: Restructuring the Score Data Set with One Observation per Student per Question

```
data restructure;
   set score;
   array Ans[*] $ 2 Ans1-Ans10;
   do Item=1 to 10;
      Choice=Ans[Item];
      output;
   end;
   keep Item Choice Percent;
run;
```

For each observation in the SCORE data set, you output 10 observations in the RESTRUCTURE data set, one observation for each item. Here are the first few observations in the RESTRUCTURE data set:

First 20 Observations in Data Set RESTRUCTURE

Percent	Item	Choice
83.9286	1	D*
83.9286	2	C*
83.9286	3	E*
83.9286	4	B*
83.9286	5	C*
83.9286	6	B*
83.9286	7	A*
83.9286	8	B*
83.9286	9	C*
83.9286	10	E*
73.2143	1	D*
73.2143	2	C*
73.2143	3	E*
73.2143	4	A
73.2143	5	C*
73.2143	6	B*
73.2143	7	A*
73.2143	8	B*
73.2143	9	A
73.2143	10	D

Using this data set, you can now compute the mean score for all students who chose each of the answers. You can use PROC MEANS to do this, using the item number and the answer choice as class variables. That is, you want to compute the mean value of Percent for each combination of Item and Class. Here is the code:

Program 5.7: Using PROC MEANS to Compute the Mean Percent for Each Combination of Item and Choice

```
proc means data=restructure mean std maxdec=2;
   class Item Choice;
   var Percent;
run;
```

The resulting output is listed next (only the first few values are displayed here):

Output from Program 5.7

The MEANS Procedure

Analysis Variable : Percent Percent score				
Item	Choice	N Obs	Mean	Std Dev
1	A	1	46.43	.
	B	2	69.64	17.68
	D*	85	77.10	7.50
	E	49	66.55	11.74
2	A	2	55.36	5.05
	B	7	70.92	5.81
	C*	112	74.51	10.51
	D	1	64.29	.
	E	14	66.84	9.75
3	B	7	63.27	15.18
	C	2	48.21	2.53
	E*	128	73.91	9.91
4	A	23	68.63	11.37
	B*	63	78.03	8.86
	C	24	68.97	11.59
	D	17	69.12	8.01
	E	9	67.46	11.39
5	A	8	61.83	15.97
	B	9	61.71	12.15
	C*	117	74.56	9.42
	D	2	80.36	2.53

In each of the first four items, the mean score of all students choosing the right answer is higher than for any other choice. However, you can see that students choosing B for item two have a mean score (70.92) almost as high as that for students choosing the correct answer (C average score = 74.51). You might want to examine choice D to see if you want to make changes.

Combining the Mean Score per Answer Choice with Frequency Counts

To make the above display even more useful, you can combine the mean score per answer choice with the answer frequencies. The best way to do this is with PROC TABULATE. This SAS procedure can combine statistical and frequency data in a single table. The PROC TABULATE statements are shown in the next program:

Program 5.8: Using PROC TABULATE to Combine Means Scores and Answer Frequencies

```
proc format;
   picture pct low-<0=' ' 0-high='009.9%';
run;

title "Displaying the Student Mean Score for Each Answer Choice";
proc tabulate data=restructure;
   class Item Choice;
   var Percent;
   table Item*Choice,
   Percent=' '*(pctn<Choice>*f=pct. mean*f=pct.
   std*f=10.2)  / rts=20 misstext=' ';
   keylabel all  = 'Total'
            mean = 'Mean Score' pctn='Freq'
            std  = 'Standard Deviation';
run;
```

The picture format in this program prints percentages to a tenth of a percent and adds the percent sign to the value. Two class variables, Item and Choice, are used to show statistics and counts for each combination of Item and Choice (the same as with PROC MEANS, described earlier). The VAR statement lists all the variables for which you want to compute statistics. Because you want to see the mean percentage score for each item and choice, you list Percent on the VAR statement. Finally, the TABLES statement defines the rows and columns in the table. We will skip some details and only indicate that the rows of the table contain values of Choice nested within Item and the columns of the table contain percentages (the keyword PCT does this), means, and standard deviations. The remainder of the statements select formats for the various values as well as labels that you want to associate with each statistic. The resulting table is shown next:

Output from Program 5.8 (Partial Listing)

Displaying the Student Mean Score for Each Answer Choice

		Freq	Mean Score	Standard Deviation
Item	Choice			
1	A	0.7%	46.4%	
	B	1.4%	69.6%	17.68
	D*	62.0%	77.1%	7.50
	E	35.7%	66.5%	11.74
2	A	1.4%	55.3%	5.05
	B	5.1%	70.9%	5.81
	C*	82.3%	74.5%	10.51
	D	0.7%	64.2%	
	E	10.2%	66.8%	9.75
3	B	5.1%	63.2%	15.18
	C	1.4%	48.2%	2.53
	E*	93.4%	73.9%	9.91
4	A	16.9%	68.6%	11.37
	B*	46.3%	78.0%	8.86
	C	17.6%	68.9%	11.59
	D	12.5%	69.1%	8.01
	E	6.6%	67.4%	11.39
[illegible]	A	5.8%	[illegible]	15.97

You now have the answer frequencies and the student mean scores for each answer choice for each item on the test in a single table. Take a look at item one. Notice that the students who answered this item correctly (answer D) had a mean test score of 77.10, which is higher than the mean test score for any of the incorrect responses.

Later sections of this chapter will build on this and add even more item information in a single table.

Computing the Proportion Correct by Quartile

Besides inspecting the point-biserial correlations (a single number), you will gain more insight into how an item is performing by dividing the class into quantiles. The number of quantiles will depend on how many students took the test—if you have a relatively small sample size, you may only want three or four quantiles. For much larger samples, you may want as many as six. Once you have divided the class into quantiles, you can then compute the proportion of students answering an item correctly in each quantile. What you hope to see is the proportion correct increasing, going from the lowest quantile to the highest. You can examine the proportions correct by quantile in tabular form or produce a graphical plot of this information. Such a plot is called an *item characteristic curve*.

In this example, you are going to use a 56-item statistics test administered to 137 students. For this sample size, dividing the class into quartiles (four groups) works well. You can use the SAS RANK procedure to divide a data set into any number of groups. Here is the first step:

Program 5.9: Dividing the Group into Quartiles

```
*Dividing the group into quartiles;
proc rank data=score(keep=Raw Score1-Score10) groups=4 out=quartiles;
   var Raw;
   ranks Quartile;
run;
```

You use the GROUPS= option to tell PROC RANK to divide the data set into four groups, based on the value of the variable Raw. The RANKS statement names the variable that will contain the group numbers. For some completely unknown reason, when you ask PROC RANK to create groups, it numbers the groups starting from 0. Because you are requesting four groups, the variable Quartile will have values from 0 to 3, as shown below:

Output from Program 5.9 (First 10 Observations)

Listing of Data Set QUARTILES

ID	Raw	Quartile	Score1	Score2	Score3	Score4	Score5
203579875	47	3	1	1	1	1	1
116841443	41	1	1	1	1	0	1
176786926	45	2	1	1	0	1	1
011413555	39	1	1	1	1	0	1
051502147	41	1	0	0	1	1	1
069456918	46	3	1	1	1	1	1
743222519	33	0	1	1	1	1	1
177458805	42	2	0	1	1	1	1
927950674	36	0	1	0	0	0	1
181246859	35	0	0	1	1	0	1

You can now compute the mean score for each quartile, like this:

Program 5.10: Computing the Mean Scores by Quartile

```
title "Mean Scores by Quartile - First 10 Items";
proc means data=quartiles mean maxdec=2;
   class Quartile;
   Var Score1-Score10;
run;
```

Here is the output:

Output from Program 5.10

Mean Scores by Quartile - First 10 Items

The MEANS Procedure

Rank for Variable Raw	N Obs	Variable	Mean
0	31	Score1	0.23
		Score2	0.61
		Score3	0.81
		Score4	0.16
		Score5	0.65
		Score6	0.87
		Score7	0.32
		Score8	0.77
		Score9	0.81
		Score10	0.71
1	35	Score1	0.69
		Score2	0.71
		Score3	0.94
		Score4	0.34
		Score5	0.91
		Score6	0.97
		Score7	0.63
		Score8	0.83
		Score9	0.80
		Score10	0.91
2	40	Score1	0.68
		Score2	0.98
		Score3	0.98
		Score4	0.53
		Score5	0.90
		Score6	1.00
		Score7	0.65
		Score8	0.83
		Score9	0.95
		Score10	0.98
3	31	Score1	0.87
		Score2	0.94
		Score3	1.00
		Score4	0.81
		Score5	0.94
		Score6	0.97
		Score7	0.94
		Score8	0.90
		Score9	1.00
		Score10	0.94

Although this listing contains the mean score for each of the four quartiles for the first 10 items on the test, the layout is inconvenient. You would rather see each item in order, with the percent correct by quartile displayed on a single line. The program to restructure and combine the answer frequencies for each of the test items, the item difficulty, the point-biserial coefficient, and the proportion correct by quartile is the subject of the next section.

Combining All the Item Statistics in a Single Table

In order to demonstrate this final program, the data from the 56-item statistics test is used. However, only the first 10 items on the test are analyzed (to reduce the size of the generated reports). The scores used to divide the class into quartiles are based on all 56 items.

The programming to combine all of the item statistics into a single table takes a number of steps and is not for the faint of heart. You can skip right to the output and its interpretation if you wish. For those readers who want to understand the programming details, the program contains callouts that link to descriptions following the program.

Program 5.11: Combining All the Item Statistics in a Single Table

```
%let Nitems=56; ❶
data score;
   infile 'c:\books\test scoring\stat_test.txt' pad;
   array Ans[&Nitems] $ 1 Ans1-Ans&Nitems;     *Student Answers;
   array Key[&Nitems] $ 1 Key1-Key&Nitems;     *Answer Key;
   array Score[&Nitems] 3 Score1-Score&Nitems; *1=right,0=wrong;
   retain Key1-Key&Nitems;
   if _n_=1 then input @11 (Key1-Key&Nitems)($1.);
   input @1  ID $9.
         @11 (Ans1-Ans&Nitems)($1.);
   do Item = 1 to &Nitems;
      Score[Item] = Key[Item] eq Ans[Item];
   end;
   Raw=sum (of Score1-Score&Nitems);
   Percent=100*Raw / &Nitems;
   keep Ans1-Ans&Nitems Key1-Key&Nitems ID Raw Percent Score1-
Score&Nitems;
   label ID      = 'Student ID'
         Raw     = 'Raw score'
         Percent = 'Percent score';
run;

*Divide the group into quartiles; ❷
proc rank data=score groups=4 out=quartiles;
   var Raw;
   ranks Quartile;
run;
```

```
*Restructure the data set so that you have one item per
 observation; ❸

data tab;
   set quartiles;
   length Choice $ 1 Item_Key $ 5;
   array Score[10] Score1-Score10;
   array ans[10] $ 1 Ans1-Ans10;
   array key[10] $ 1 Key1-Key10;
   Quartile = Quartile + 1;
   do Item=1 to 10;
      Item_Key = cat(right(put(Item,3.))," ",Key[Item]);
      Correct=Score[Item];
      Choice=Ans[Item];
      output;
   end;
   keep Item Item_Key Quartile Correct Choice;
run;

*Sort by item number; ❹
proc sort data=tab;
   by Item;
run;

*Compute correlation coefficients; ❺
proc corr data=score nosimple noprint
          outp=corrout(where=(_type_='CORR'));
   var Score1-Score10;
   with Raw;
run;

*Restructure correlation data set; ❻
data corr;
   set corrout;
   array Score[10];
   do Item=1 to 10;
      Corr = Score[Item];
      output;
   end;
   keep Item Corr;
run;

*Combine correlations and quartile information; ❼
data both;
   merge corr tab;
   by Item;
run;
```

```
*Print out final table;
proc tabulate format=7.2 data=both order=internal noseps;
   title "Item Statistics";
   label Quartile = 'Quartile'
         Choice   = 'Choices';
   class Item_Key Quartile Choice;
   var Correct Corr;
   table Item_Key = 'Num Key'*f=6. ,
      Choice*(pctn<Choice>)*f=3.
      Correct=' '*mean='Diff.'*f=percent5.2
      Corr=' '*mean='Corr.'*f=5.2
      Correct=' '*Quartile*mean='Prop. Correct'*f=percent7.2/
      rts=8;
   keylabel pctn='%' ;
run;
```

❶ You start out by scoring the test in the usual way. Instead of hard coding the number of items on the test, you use the macro variable &Nitems and assign the number of items (56) with a %LET statement. Using %LET is another way of assigning a value to a macro variable. There is one other small difference in this program compared to the scoring programs displayed previously: The variables in the Score array are stored in 3 bytes (notice the 3 before the list of variables in this array). Since the values of the Score variables are 0s and 1s, you do not need to store them in 8 bytes (the default storage length for SAS numeric values). Three bytes is the minimum length allowed by SAS.

❷ You use PROC RANK to create a variable (that you call Quartile) that represents quartiles of the Raw score. Remember that the values of Quartile range from 0 to 3. Below are the first few observations from data set QUARTILES (with selected variables displayed):

First Few Observations in Data Set Quartiles (Selected Variables Displayed)

Listing of Data Set QUARTILES

Raw	Quartile	Ans1	Ans2	Key1	Key2	Score1	Score2
47	3	D	C	D	C	1	1
41	1	D	C	D	C	1	1
45	2	D	C	D	C	1	1
39	1	D	C	D	C	1	1
41	1	E	E	D	C	0	0
46	3	D	C	D	C	1	1
33	0	D	C	D	C	1	1
42	2	E	C	D	C	0	1
36	0	D	E	D	C	1	0

❸ You need to restructure this data set so that there is one observation per test item. That way it can be combined with the correlation data and later displayed in a table with one item per row. In this DATA step, you accomplish several goals. First, you add 1 to Quartile so that the values now range from 1 to 4 (instead of 0 to 3). Next, you create a new variable (Item_Key) that puts together (*concatenates*, in computer jargon) the item number and the answer key. Each of the item scores (the 0s and 1s) is assigned to the variable Correct. Finally, each answer choice is assigned to a variable you call Choice.

❹ You sort the TAB data set by item (so that you can combine it with the correlation data). Here are the first 20 observations from the sorted data set. Looking at this listing should help you understand step 3:

First 20 Observations from Data Set TAB after Sorting

Listing of Data Set TAB

Quartile	Choice	Item_Key	Item	Correct
4	D	1D	1	1
2	D	1D	1	1
3	D	1D	1	1
2	D	1D	1	1
2	E	1D	1	0
4	D	1D	1	1
1	D	1D	1	1
3	E	1D	1	0
1	D	1D	1	1
1	E	1D	1	0
4	D	1D	1	1
4	E	1D	1	0
3	D	1D	1	1
3	D	1D	1	1
4	E	1D	1	0
1	E	1D	1	0
3	D	1D	1	1
3	D	1D	1	1
4	D	1D	1	1
4	D	1D	1	1

The mean of the variable Correct for Item would be the proportion of the entire class that answered the item correctly. If you compute the mean for each of the four quartiles of the class, you have the proportion correct by quartile, one of the values you want to display in the final table.

❺ You use PROC CORR to compute the point-biserial correlations for each of the items and place these correlations in a data set called CORROUT. Here is the listing of this data set:

Listing of Data Set CORROUT

TYPE	_NAME_	Score1	Score2	Score3	Score4	Score5	Score6	Score7	Score8	Score9	Score10
CORR	Raw	0.48830	0.29773	0.32225	0.43235	0.35183	0.20284	0.41984	0.065315	0.27607	0.38473

❻ You now restructure the correlation data set so that there is one observation per item. Here is the listing:

Listing of Data Set CORR

Item	Corr
1	0.48830
2	0.29773
3	0.32225
4	0.43235
5	0.35183
6	0.20284
7	0.41984
8	0.06532
9	0.27607
10	0.38473

❼ You can now combine the data from the two data sets (TAB and CORR) since they are now in Item order.

❽ You use PROC TABULATE to compute the mean value of Correct for each of the quartiles and display all the statistics for each item. PROC TABULATE allows you to format each of the cells in the table. After all this work, here is the final table:

Item Statistics

	Choices							Quartile			
	A	B	C	D	E			1	2	3	4
	%	%	%	%	%	Diff.	Corr.	Prop. Correct	Prop. Correct	Prop. Correct	Prop. Correct
Num Key											
1 D	1	1	.	62	36	62%	0.49	22.6%	68.6%	67.5%	87.1%
2 C	1	5	82	1	10	82%	0.30	63.3%	71.4%	97.5%	93.5%
3 E	.	5	1	.	93	93%	0.32	80.6%	94.3%	97.5%	100%
4 B	17	46	18	13	7	46%	0.43	16.1%	35.3%	52.5%	80.6%
5 C	6	7	85	1	1	85%	0.35	64.5%	91.4%	90.0%	93.5%
6 B	1	96	3	.	.	96%	0.20	87.1%	97.1%	100%	96.8%
7 A	64	21	6	1	8	64%	0.42	32.3%	62.9%	65.0%	93.5%
8 B	7	84	.	4	5	84%	0.07	77.4%	82.9%	82.5%	93.3%
9 C	9	.	89	1	1	89%	0.28	80.6%	80.0%	95.0%	100%
10 E	.	3	4	4	89	89%	0.38	71.0%	91.4%	97.5%	93.5%

Yes, that was a lot of work, but you can now see all of the item information (the percent of the class choosing each of the answer choices, the item difficulty, the point-biserial coefficient, and the proportion correct by quartile) in a single table. SAS macros (pre-packaged programs) to accomplish all the tasks described in this book can be found in Chapter 11.

Interpreting the Item Statistics

The definition of a "good" item depends somewhat on why you are testing people in the first place. If the test is designed to rank students by ability in a class, you would like items with high point-biserial correlations and an increasing value of the percent-by-quartile statistic. For example, take a look at item 4. Notice that all of the answer choices have been selected. This indicates that there are no obvious wrong answers that all the students can reject. Next, notice that this is a fairly difficult item—only 46% of the students answered this item correctly. As a teacher, you may find this disappointing, but as a psychometrician, you are pleased that this item has a fairly high point-biserial correlation (.43) and the proportion of the students answering the item correctly increases from 16.1% to 80.6% over the four quartiles.

Let's look at another item. Most students answered item 3 correctly (difficulty = 93%). Items that are very easy or very hard are not as useful in discriminating student ability as other items. Item 3, although very easy, still shows an increase in the proportion by quartile but, because this increase is not as dramatic as item 4, the point-biserial correlation is a bit lower. Suppose you had an item that every student answered correctly. Obviously, this item would not be useful in discriminating good students from poor students (the point-biserial correlation would be 0). If your goal is to determine whether students understand certain course objectives, you may decide that it is OK to keep items that almost all students answer correctly.

Conclusion

Each of us has taken tests with poorly written items. We see a question that uses words like "every" or "never." You can think of one or two really rare exceptions and wonder: Is the teacher trying to trick me or am I reading into the question? Not only are poorly written items frustrating to the test taker, but they also reduce the test's reliability. Using the methods and programs described in this chapter is a good first step in identifying items that need improvement.

Index

A

achievement tests
  about 117
  multiple-choice (MC) items 118–121
American Educational Research Association 2
American Psychological Association 2
analysis, classical item
  combining item statistics in single tables 66–72
  combining mean score with frequency counts 61–63
  computing proportion correct by quartile 63–66
  displaying mean score 55–60
  interpreting item statistics 72–73
  point-biserial correlation coefficient 51–53
  producing attractive reports 53–54
  restructuring data set 54–55
  on 30-item physics test 112
analyzing tests with multiple versions 80–86
answer frequencies
  about 31
  developing automated programs to score tests and produce item frequencies 34–36
  displaying in graphical form 36–40
  displaying in tabular form 31–32
  modifying programs to display answers in frequency tables 33–34
answer keys, reading from separate files 16–17
ARRAY statements 12, 33, 81
assessing test reliability
  See test reliability

B

bar charts, producing 39–40
Birnbaum, Allan 98
BY statement 25

C

Cartesian products, creating 46
cheating
  detecting on multiple-choice tests 123–140
  programs to detect 163–170
  programs to search for possible 170–173
classical item analysis
  combining item statistics in single tables 66–72
  combining mean score with frequency counts 61–63
  computing proportion correct by quartile 63–66
  displaying mean score 55–60
  interpreting item statistics 72–73
  point-biserial correlation coefficient 51–53
  producing attractive reports 53–54
  restructuring data set 54–55
  on 30-item physics test 112
classical test theory (CTT), compared with Item Response Theory (IRT) 99–100
Cody, Ron
  Learning SAS By Example: A Programmer's Guide 3
COLUMNS statement 27
combining
  item statistics in single tables 66–72
  mean score with frequency counts 61–63
comma-delimited data (CSV file), reading 12–13
computing
  correlations between odd/even scores 90
  Cronbach's Alpha 93
  Kuder-Richardson Formula 20 (KR-20) 92–93, 161–162
  mean score 57–58, 134
  proportion correct by quartile 63–66
  Spearman-Brown Adjusted Split-Half Correlation 90–91
  split-half reliability 88–91

standard deviation 134
constructed response 119
CORR procedure
computing biserial correlations 71
computing correlations between odd/even scores 90
computing Cronbach's Alpha 93
computing Spearman-Brown Adjusted Split-Half Correlation 91
point-biserial correlation coefficient 51–53
producing attractive reports 53
CORRESPONDANCE array 81–82
creating
Cartesian products 46
rosters using REPORT procedure 27
Cronbach's Alpha, computing 93
CSV (comma-delimited) file, reading 12–13
CTT (classical test theory), compared with Item Response Theory (IRT) 99–100

D

data
comma-delimited 12–13
reading directly from Excel workbooks 14–16
reading from text files 6–10
space-delimited 10–12
data checking program 150–151
data sets
preparing for IRT procedure 102–103
restructuring 54–55
DBMS= option 28
DEFINE statement 27
deleting items from scoring programs 78–80
Delwiche, Laura
The Little SAS Book: A Primer, Fifth Edition 3
demonstrating
effect of item discrimination on test reliability 94–95
effect of test length on test reliability 95
detecting
cheating on multiple-choice tests 123–140
invalid IDs and answer choices 42–43
developing automated programs to score tests and produce item frequencies 34–36
displaying
answer frequencies in graphical form 36–40
answer frequencies in tabular form 31–32
histograms of test scores 20–24
mean score 55–60
DO loop 9, 79
duplicate records, checking for and eliminating 47–48

E

eliminating duplicate records 47–48
error checking
detecting invalid IDs and answer choices 42–43
duplicate records 47–48
for ID errors 43–45
identifying invalid IDs with "fuzzy" matching 45–47
test data 41–49
Excel files
analyzing tests with multiple versions 82–86
exporting student rosters to 28
reading data directly from workbooks 14–16
scoring multiple test versions from 157–161
scoring tests from 146–148
EXPORT procedure 28
exporting student rosters to Excel 28

F

FILE PRINT statement 43
files
  Excel 14–16, 82–86, 146–148, 157–161
  reading answer keys from separate 16–17
  text 6–10, 144–145, 155–157
FINDC function 77
FREQ procedure
  determining answer frequencies 56–57
  displaying answer frequencies 31–32
  inspecting variable score 104–105
  modifying programs to display answers in frequency tables 34
frequency counts, combining with mean score 61–63
frequency tables, modifying programs to display answers in 33–34
functions
  See specific functions
"fuzzy" matching 45–47

G

graphical form, displaying answer frequencies in 36–40
GROUPS= option 64

H

HBAR statement 36, 40
HISTOGRAM statement 20, 24
histograms
  displaying of test scores 20–24
  plotting 134
  producing 128

I

ICC (item characteristic curve) 100
IDs
  error checking 43–45
  identifying invalid with "fuzzy" matching 45–47
  matching student names with 24–26
indirect addressing 82
INFILE statement 9, 12, 17, 24, 44, 77, 82
INPUT statement 12, 77
interpreting item statistics 72–73
IRT procedure, running 104–110
  See also Item Response Theory (IRT)
item analysis
  about 1–2
  program for 151–153
item baggage 120
item characteristic curve (ICC) 100
item discrimination, demonstrating effect of on test reliability 94–95
item frequencies, producing 34–36
Item Response Theory (IRT)
  about 97–99
  advanced 102
  basics of 99–100
  classical item analysis on 30-item physics test 112
  compared with classical test theory (CTT) 99–100
  preparing data sets for IRT procedure 102–103
  results 100–101
  running IRT procedure 104–110
  running other models 111
item statistics
  combining in single tables 66–72
  interpreting 72–73

J

joint-wrongs 123, 131

K

Kane, M.T.
  "Validation" 2
KEEP statement 89
Kuder-Richardson Formula 20 (KR-20), computing 92–93, 161–162
kurtosis statistic 21

L

LABEL statement 10
latent trait theory
    See Item Response Theory (IRT)
Learning SAS By Example: A Programmer's Guide (Cody) 3
LENGTH statement 12, 25
%LET statement 68, 79
LIBNAME statement 15, 86
libref 15
list input 11
The Little SAS Book: A Primer, Fifth Edition (Delwiche and Slaughter) 3
Lord, Frederick 98

M

matching student names with student IDs 24–26
mean score
    combining with frequency counts 61–63
    computing 134
    displaying 55–60
MEANS procedure
    computing Kuder-Richardson Formula 20 (KR-20) 92–93
    computing mean score 57–58, 134
    computing proportion correct by quartile 65
    computing standard deviation 134
MERGE statement 26
MISSOVER option 12
MODEL statement 111
modern test theory
    See Item Response Theory (IRT)
modifying
    programs to display answers in frequency tables 33–34
    programs to score tests of arbitrary numbers of items 17–19
    scoring program to accept alternate correct answers 76–78
multiple-choice (MC) items, writing
    about 115, 118–120
    achievement tests 117–121
    getting organized 116–117
    taxonomy of objectives and items 116–117
    test blueprints 116
multiple-choice tests
    detecting cheating on 123–140
    searching for matches 136–140

N

named literal 15
National Council on Measurement in Education 2
NOCUM option 32
NOPRINT option 53
NOTDIGIT function 43

O

one-parameter model (1PL) 98
options
    See specific options
OUTFILE= option 28

P

PLOTS= procedure 105
plotting histograms 134
point-biserial correlation coefficient 51–53
preparing data sets for IRT procedure 102–103
PRINT procedure 10, 19
printing rosters 148–149
procedures
    See specific procedures
producing
    attractive reports 53–54
    bar charts 39–40
    histograms 128
    item frequencies 34–36
programs
    See also scoring program
    data checking 150–151
    to delete items 154–155
    to detect cheating 163–170

developing to score tests and produce item frequencies 34–36
item analysis 151–153
modifying to display answers in frequency tables 33–34
modifying to score tests of arbitrary numbers of items 17–19
to rescore tests 154–155
to search for possible cheating 170–173
PUT statement 43, 86, 91

Q

quartile, computing proportion correct by 63–66

R

RANK procedure 63–64, 68
Rasch, George 98
Rasch model 98
reading
answer keys from separate files 16–17
comma-delimited data (CSV file) 12–13
data directly from Excel workbooks 14–16
data from text files 6–10
space-delimited data 10–12
recognition format 117
reliability
See test reliability
REPLACE option 28
REPORT procedure 27
reports, producing attractive 53–54
%RESCORE macro 94
rescoring tests 78–80, 154–155
RESFUNC= option 111
restructuring data sets 54–55
RETAIN statement 9
Review of Research in Education (Shepard) 2
rosters
creating using REPORT procedure 27
exporting to Excel 28
printing 148–149
RUN statement 10
running
IRT procedure 104–110
other models 111

S

SAS 2–3
See also specific topics
SAS 9.4 Language Reference: Concepts 3
SAS 9.4 Macro Language Reference 3
SCAN function 24
%SCORE macro 20
scoring
analyzing tests with multiple versions 82–84
multiple test versions from Excel files 157–161
multiple test versions from text files 155–157
tests 6–10, 34–36, 144–148
tests from Excel files 146–148
tests from text files 144–145
scoring program
about 75
analyzing tests with multiple versions 80–82, 84–86
deleting items 78–80
modifying to accept alternate correct answers 76–78
rescoring test 78–80
scree plot approach 108
searching for matches in multiple-choice tests 136–140
SET statement 15
SGPLOT procedure
displaying answer frequencies in graphical form 36
plotting histograms 134
producing bar charts 39–40
producing histograms 128
SHEET= option 28
Shepard, L.A.
Review of Research in Education 2
skewness 21

Slaughter, Susan
The Little SAS Book: A Primer, Fifth Edition 3
SORT procedure 10, 25, 38, 47–48
space-delimited data, reading 10–12
Spearman-Brown Adjusted Split-Half Correlation, computing 90–91
Spearman-Brown formula 88
SPEDIS function 46–47
split-half reliability, computing 88–91
SQL procedure 46
standard deviation, computing 134
Standards for educational and psychological testing 2
statements
See specific statements
student names, matching with student IDs 24–26
SUBSTR function 33–34
swap-and-drop operation 86

T

tables, combining item statistics in single 66–72
TABLES statement 32, 61
tabular form, displaying answer frequencies in 31–32
TABULATE procedure
combining mean score with frequency counts 61
computing mean value 72
taxonomy, of objectives and items 116–117
test reliability
about 1–2, 87
computing Cronbach's Alpha 93
computing Kuder-Richardson Formula 20 (KR-20) 92–93
computing split-half reliability 88–91
demonstrating effect of item discrimination on 94–95
demonstrating effect of test length on 95
test-retest reliability 87
tests
achievement 117–121
analyzing with multiple versions 80–86
blueprints for 116
demonstrating effect of length of on test reliability 95
displaying histograms of scores 20–24
error checking data 41–49
multiple-choice (MC) 115–120, 123–140
rescoring 78–80, 154–155
scoring 6–10, 34–36, 144–148
scoring from Excel files 146–148
scoring from text files 144–145
text files
analyzing tests with multiple versions 80–82
reading data from 6–10
scoring multiple test versions from 155–157
scoring tests from 144–145
three-parameter model (3PL) 98
See also Item Response Theory (IRT)
TRANSPOSE procedure 54–55
two-parameter model (2PL) 98–99
See also Item Response Theory (IRT)

U

unidimensionality 98
UNIVARIATE procedure 23

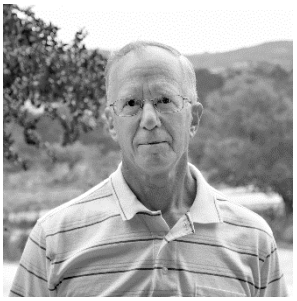
V

"Validation" (Kane) 2
VAR statement 20, 52, 61
VBAR statement 36

W

WITH statement 52
workbooks (Excel), reading data directly from 14–16
Wright, Benjamin 98
writing multiple-choice (MC) items
See multiple-choice (MC) items, writing

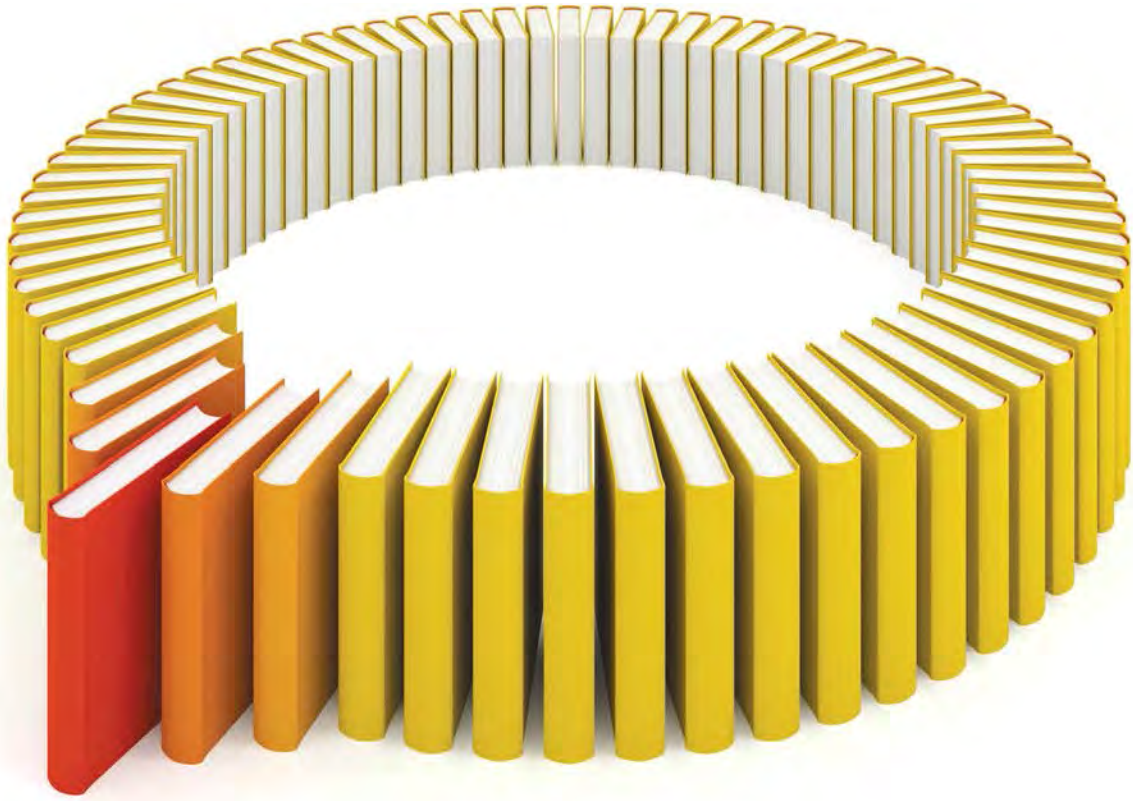
About These Authors



Ron Cody, EdD, a retired professor from the Robert Wood Johnson Medical School now works as a private consultant and a national instructor for SAS Institute Inc. A SAS user since 1977, Ron's extensive knowledge and innovative style have made him a popular presenter at local, regional, and national SAS conferences. He has authored or co-authored numerous books, such as *Learning SAS by Example: A Programmer's Guide; SAS Statistics by Example, Applied Statistics and the SAS Programming Language, Fifth Edition; The SAS Workbook; The SAS Workbook Solutions; Cody's Data Cleaning Techniques Using SAS, Second Edition; Longitudinal Data and SAS: A Programmer's Guide; SAS Functions by Example, Second Edition*, and *Cody's Collection of Popular Programming Tasks and How to Tackle Them*, as well as countless articles in medical and scientific journals.

Jeffrey Smith, is professor and Associate Dean (Research) in the College of Education at the University of Otago in New Zealand. For 29 years he was on the faculty of Rutgers University, serving the Department of Educational Psychology as professor and chair. From 1988 to 2005 he also served as Head of the Office of Research and Evaluation at the Metropolitan Museum of Art. He has written or edited eight books on educational assessment and statistics, the psychology of aesthetics, and educational psychology. He has published more than 70 research articles and reviews in the field of education, also founding and co-editing a journal, *Psychology of Aesthetics, Creativity, and the Arts*. Smith received his undergraduate degree from Princeton University and his Ph.D. from the University of Chicago.

Learn more about these authors by visiting their author pages, where you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more:
http://support.sas.com/publishing/authors/cody.html
http://support.sas.com/publishing/authors/smith_jeff.html



Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore
for additional books and resources.

sas
THE POWER TO KNOW®