SAS/STAT® 13.2 User's Guide

The SCORE Procedure

This document is an individual chapter from *SAS/STAT® 13.2 User's Guide*.

The correct bibliographic citation for the complete manual is as follows: SAS Institute Inc. 2014. *SAS/STAT® 13.2 User's Guide*. Cary, NC: SAS Institute Inc.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

August 2014

Chapter 88
The SCORE Procedure

Contents

Overview: SCORE Procedure

The SCORE procedure multiplies values from two SAS data sets, one containing coefficients (for example, factor-scoring coefficients or regression coefficients) and the other containing raw data to be scored using the coefficients from the first data set. The result of this multiplication is a SAS data set containing linear combinations of the coefficients and the raw data values.

Many statistical procedures output coefficients that PROC SCORE can apply to raw data to produce scores. The new score variable is formed as a linear combination of raw data and scoring coefficients. For each observation in the raw data set, PROC SCORE multiplies the value of a variable in the raw data set by the matching scoring coefficient from the data set of scoring coefficients. This multiplication process is repeated for each variable in the VAR statement. The resulting products are then summed to produce the value of the new score variable. This entire process is repeated for each observation in the raw data set. In other words, PROC SCORE cross multiplies part of one data set with another.

Raw Data Set

The raw data set can contain the original data used to calculate the scoring coefficients, or it can contain an entirely different data set. The raw data set must contain all the variables needed to produce scores. In addition, the scoring coefficients and the variables in the raw data set that are used in scoring must have the same names. See the section "Getting Started: SCORE Procedure" on page 7295 for more information.

Scoring Coefficients Data Set

The data set containing scoring coefficients must contain two special variables: the _TYPE_ variable and the _NAME_ or _MODEL_ variable.

- The _TYPE_ variable identifies the observations that contain scoring coefficients.
- The _NAME_ or _MODEL_ variable provides a SAS name for the new score variable.

PROC SCORE first looks for a _NAME_ variable in the SCORE= input data set. If there is such a variable, the variable's value is what SCORE uses to name the new score variable. If the SCORE= data set does not have a _NAME_ variable, then PROC SCORE looks for a _MODEL_ variable.

For example, PROC FACTOR produces an output data set that contains factor-scoring coefficients. In this output data set, the scoring coefficients are identified by _TYPE_='SCORE'. For _TYPE_='SCORE', the _NAME_ variable has values of 'Factor1', 'Factor2', and so forth. PROC SCORE gives the new score variables the names Factor1, Factor2, and so forth.

As another example, the REG procedure produces an output data set that contains parameter estimates. In this output data set, the parameter estimates are identified by _TYPE_='PARMS'. The _MODEL_ variable contains the label used in the MODEL statement in PROC REG, or it uses MODEL*n* if no label is specified. This label is the name PROC SCORE gives to the new score variable.

Standardization of Raw Data

PROC SCORE automatically standardizes or centers the DATA= variables for you, based on information from the original variables and analysis from the SCORE= data set.

If the SCORE= scoring coefficients data set contains observations with _TYPE_='MEAN' and _TYPE_='STD', then PROC SCORE standardizes the raw data before scoring. For example, this type of SCORE= data set can come from PROC PRINCOMP without the COV option.

If the SCORE= scoring coefficients data set contains observations with _TYPE_='MEAN' but _TYPE_='STD' is absent, then PROC SCORE centers the raw data (the means are subtracted) before scoring. For example, this type of SCORE= data set can come from PROC PRINCOMP with the COV option.

If the SCORE= scoring coefficients data set does not contain observations with _TYPE_='MEAN' and _TYPE_='STD', or if you use the NOSTD option, then PROC SCORE does not center or standardize the raw data.

If the SCORE= scoring coefficients are obtained from observations with _TYPE_='USCORE', then PROC SCORE "standardizes" the raw data by using the uncorrected standard deviations identified by _TYPE_='USTD', and the means are not subtracted from the raw data. For example, this type of SCORE= data set can come from PROC PRINCOMP with the NOINT option. For more information about _TYPE_='USCORE' scoring coefficients in TYPE=UCORR or TYPE=UCOV output data sets, see Appendix A, "Special SAS Data Sets."

You can use PROC SCORE to score the data that were also used to generate the scoring coefficients, although more typically, scoring results are directly obtained from the OUT= data set in a procedure that computes scoring coefficients. When scoring new data, it is important to realize that PROC SCORE assumes that the new data have approximately the same scales as the original data. For example, if you specify the COV option with PROC PRINCOMP for the original analysis, the scoring coefficients in the PROC PRINCOMP OUTSTAT= data set are not appropriate for standardized data. With the COV option, PROC PRINCOMP will not output _TYPE_='STD' observations to the OUTSTAT= data set, and PROC SCORE will only subtract the means of the original (not new) variables from the new variables before multiplying. Without the COV option in PROC PRINCOMP, both the original variable means and standard deviations will be in the OUTSTAT= data set, and PROC SCORE will subtract the original variable means from the new variables and divide them by the original variable standard deviations before multiplying.

In general, procedures that output scoring coefficients in their OUTSTAT= data sets provide the necessary information for PROC SCORE to determine the appropriate standardization. However, if you use PROC SCORE with a scoring coefficients data set that you constructed without _TYPE_='MEAN' and _TYPE_='STD' observations, you might have to do the relevant centering or standardization of the new data first. If you do this, you must use the means and standard deviations of the original variables—that is, the variables that were used to generate the coefficients—not the means and standard deviations of the variables to be scored.

See the section "Getting Started: SCORE Procedure" on page 7295 for further illustration.

Getting Started: SCORE Procedure

The SCORE procedure multiplies the values from two SAS data sets and creates a new data set to contain the results of the multiplication. The variables in the new data set are linear combinations of the variables in the two input data sets. Typically, one of these data sets contains raw data that you want to score, and the other data set contains scoring coefficients.

The following example demonstrates how to use the SCORE procedure to multiply values from two SAS data sets, one containing factor-scoring coefficients and the other containing raw data to be scored using the scoring coefficients.

Suppose you are interested in the performance of three different types of schools: private schools, state-run urban schools, and state-run rural schools. You want to compare the schools' performances as measured by student grades on standard tests in English, mathematics, and biology. You administer these tests and record the scores for each of the three types of schools.

The following DATA step creates the SAS data set Schools. The data are provided by Chaseling (1996).

```
data Schools;
   input Type $ English Math Biology @@;
   datalines;
p  52  55  45  p  42  49  40  p  63  64  54
p  47  50  51  p  64  69  47  p  63  67  54
p  59  63  42  p  56  61  41  p  41  44  72
p  39  42  45  p  56  63  44  p  63  73  42

   ... more lines ...

r  50  47  49  r  55  48  46  r  38  36  51
;
```

The data set Schools contains the character variable Type, which represents the type of school. Valid values are p (private schools), r (state-run rural schools), and u (state-run urban schools).

The three numeric variables in the data set are English, Math, and Biology, which represent the student scores for English, mathematics, and biology, respectively. The double trailing at sign (@@) in the INPUT statement specifies that observations are input from each line until all values are read.

The following statements invoke the FACTOR procedure to compute the data set of factor scoring coefficients. The statements perform a principal components factor analysis that uses all three numeric variables in the SAS data set Schools. The OUTSTAT= option requests that PROC FACTOR output the factor scores to the data set Scores. The NOPRINT option suppresses display of the output.

```
proc factor data=Schools score outstat=Scores noprint;
   var english math biology;
run;

proc score data=schools score=Scores out=New;
   var english math biology;
   id type;
run;
```

The SCORE procedure is then invoked using Schools as the raw data set to be scored and Scores as the scoring data set. The OUT= option creates the SAS data set New to contain the linear combinations.

The VAR statement specifies that the variables English, Math, and Biology are used in computing scores. The ID statement copies the variable Type from the Schools data set to the output data set New.

The following statements print the SAS output data set Scores, the first two observations from the original data set Schools, and the first two observations of the resulting data set New.

```
title 'OUTSTAT= Data Set from PROC FACTOR';
proc print data=Scores;
run;

title 'First Two Observations of the DATA= Data Set from PROC SCORE';
proc print data=Schools(obs=2);
run;

title 'First Two Observations of the OUT= Data Set from PROC SCORE';
proc print data=New(obs=2);
run;
```

Figure 88.1 displays the output data set Scores produced by the FACTOR procedure. The last observation (number 11) contains the scoring coefficients (_TYPE_='SCORE'). Only one factor has been retained.

Figure 88.1 Listing of the Data Set Created by PROC FACTOR

OUTSTAT= Data Set from PROC FACTOR

Obs	_TYPE_	_NAME_	English	Math	Biology
1	MEAN		55.525	52.325	50.350
2	STD		12.949	12.356	12.239
3	N		120.000	120.000	120.000
4	CORR	English	1.000	0.833	0.672
5	CORR	Math	0.833	1.000	0.594
6	CORR	Biology	0.672	0.594	1.000
7	COMMUNAL		0.881	0.827	0.696
8	PRIORS		1.000	1.000	1.000
9	EIGENVAL		2.405	0.437	0.159
10	PATTERN	Factor1	0.939	0.910	0.834
11	SCORE	Factor1	0.390	0.378	0.347

Figure 88.2 lists the first two observations of the original SAS data set (Schools).

Figure 88.2 First Two Observations of the Schools Data Set

First Two Observations of the DATA= Data Set from PROC SCORE

Obs	Type	English	Math	Biology
1	p	52	55	45
2	p	42	49	40

Figure 88.3 lists the first two observations of the output data set New created by PROC SCORE.

Figure 88.3 Listing of the New Data Set

First Two Observations of the OUT= Data Set from PROC SCORE

Obs	Type	Factor1
1	p	-0.17604
2	p	-0.80294

The score variable Factor1 in the New data set is named according to the value of the _NAME_ variable in the Scores data set. The values of the variable Factor1 are computed as follows: the DATA= data set variables are standardized using the same means and standard deviations that PROC FACTOR used when extracting the factors because the Scores data set contains observations with _TYPE_='MEAN' and _TYPE_='STD'.

Note that in order to correctly use standardized scoring coefficients created by other procedures such as PROC FACTOR in this example, the data to be scored must be standardized in the same way that the data were standardized when the scoring coefficients were computed. Otherwise, the resulting scores might be incorrect. PROC SCORE does this automatically if the SCORE= data set is the original OUTSTAT= data set output from the procedure creating the scoring coefficients.

These standardized variables are then multiplied by their respective standardized scoring coefficients from the data set Scores. These products are summed over all three variables, and the sum is the value of the new variable Factor1. The first two values of the scored variable Factor1 are obtained as follows:

$$\left(\frac{(52 - 55.525)}{12.949} \times 0.390\right) + \left(\frac{(55 - 52.325)}{12.356} \times 0.378\right) + \left(\frac{(45 - 50.350)}{12.239} \times 0.347\right) = -0.17604$$

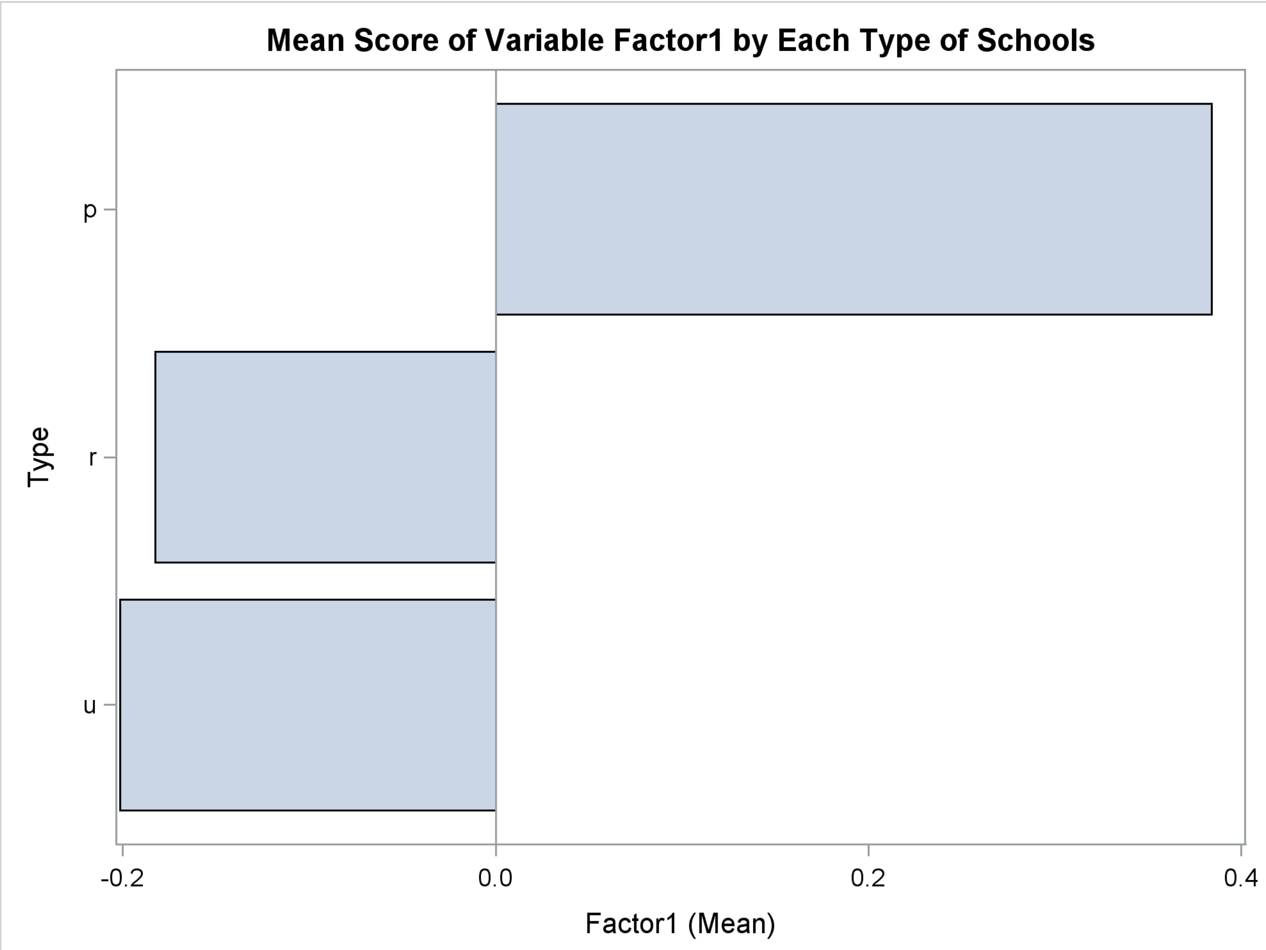
$$\left(\frac{(42 - 55.525)}{12.949} \times 0.390\right) + \left(\frac{(49 - 52.325)}{12.356} \times 0.378\right) + \left(\frac{(40 - 50.350)}{12.239} \times 0.347\right) = -0.80294$$

The following statements request that the SGPLOT procedure produce a horizontal bar chart of the variable Type. The length of each bar represents the mean of the variable Factor1.

```
title 'Mean Score of Variable Factor1 by Each Type of Schools';
proc sgplot data=New;
   hbar type / stat = mean response=Factor1;
run;
```

Figure 88.4 displays the mean score of the variable Factor1 for each of the three school types. For private schools (Type=p), the average value of the variable Factor1 is 0.384, while for state-run schools the average values are much lower. The state-run urban schools (Type=u) have the lowest mean value of –0.202, and the state-run rural schools (Type=r) have a mean value of –0.183.

Figure 88.4 Bar Chart of School Type



Syntax: SCORE Procedure

The following statements are available in the SCORE procedure:

PROC SCORE DATA=*SAS-data-set* < *options* > **;**
  BY *variables* **;**
  ID *variables* **;**
  VAR *variables* **;**

The only required statement is the PROC SCORE statement. The BY, ID, and VAR statements are described following the PROC SCORE statement.

PROC SCORE Statement

PROC SCORE DATA=*SAS-data-set* < *options* > **;**

The PROC SCORE statement invokes the SCORE procedure. Table 88.1 summarizes the options available in the PROC SCORE statement.

Table 88.1 PROC SCORE Statement Options

Option	Description
DATA=	Names the input SAS data set containing the raw data to score
NOSTD	Suppresses centering and scaling of the raw data
OUT=	Specifies the name of the SAS data set
PREDICT	Treats coefficients of –1 as 0
RESIDUAL	Reverses the sign of each score
SCORE=	Names the data set containing the scoring coefficients
TYPE=	Specifies the observations that contain scoring coefficients

You can specify the following options in the PROC SCORE statement.

DATA=*SAS-data-set*

names the input SAS data set containing the raw data to score. This option is required.

NOSTD

suppresses centering and scaling of the raw data. Ordinarily, if PROC SCORE finds _TYPE_='MEAN', _TYPE_= 'USCORE', _TYPE_='USTD', or _TYPE_='STD' observations in the SCORE= data set, the procedure uses these to standardize the raw data before scoring.

OUT=*SAS-data-set*

specifies the name of the SAS data set created by PROC SCORE. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Language Reference: Concepts*. If the OUT= option is omitted, PROC SCORE still creates an output data set and automatically names it according to the DATA*n* convention, as if you omitted a data set name in a DATA statement.

PREDICT

specifies that PROC SCORE should treat coefficients of –1 in the SCORE= data set as 0. In regression applications, the dependent variable is coded with a coefficient of –1. Applied directly to regression results, PROC SCORE produces negative residuals (see the description of the RESIDUAL option, which follows); the PREDICT option produces predicted values instead.

RESIDUAL

reverses the sign of each score. Applied directly to regression results, PROC SCORE produces negative residuals (PREDICT–ACTUAL); the RESIDUAL option produces positive residuals (ACTUAL–PREDICT) instead.

SCORE=*SAS-data-set*

names the data set containing the scoring coefficients. If you omit the SCORE= option, the most recently created SAS data set is used. This data set must have two special variables: _TYPE_ and either _NAME_ or _MODEL_.

TYPE=*name* **or** *'string'*

specifies the observations in the SCORE= data set that contain scoring coefficients. The TYPE= procedure option is unrelated to the data set option that has the same name. PROC SCORE examines the values of the special variable _TYPE_ in the SCORE= data set. When the value of _TYPE_ matches TYPE=*name*, the observation in the SCORE= data set is used to score the raw data in the DATA= data set.

If you omit the TYPE= option, scoring coefficients are read from observations with either _TYPE_='SCORE' or _TYPE_='USCORE'. Because the default for PROC SCORE is TYPE=SCORE, you need not specify the TYPE= option for factor scoring or for computing scores from OUTSTAT= data sets from the CANCORR, CANDISC, PRINCOMP, or VARCLUS procedure. When you use regression coefficients from PROC REG, specify TYPE=PARMS.

The maximum length of the argument specified in the TYPE= option depends on the length defined by the VALIDVARNAME= SAS system option. For additional information, see *SAS System Options: Reference*.

Note that the TYPE= option setting is not case sensitive. For example, the two option settings TYPE='MyScore' and TYPE='myscore' are equivalent.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC SCORE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement for the SCORE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the discussion in *SAS Language Reference: Concepts*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

You can specify a BY statement to apply separate groups of scoring coefficients to the entire DATA= data set.

If the DATA= data set does not contain any of the BY variables, the entire DATA= data set is scored by each BY group of scoring coefficients in the SCORE= data set.

If the DATA= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the DATA= data set as in the SCORE= data set, then PROC SCORE prints an error message and stops.

If all the BY variables appear in the DATA= data set with the same type and length as in the SCORE= data set, then each BY group in the DATA= data set is scored using scoring coefficients from the corresponding BY group in the SCORE= data set. The BY groups in the DATA= data set must be in the same order as in the SCORE= data set. All BY groups in the DATA= data set must also appear in the SCORE= data set. If you do not specify the NOTSORTED option, some BY groups can appear in the SCORE= data set but not in the DATA= data set; such BY groups are not used in computing scores.

ID Statement

ID *variables* **;**

The ID statement identifies variables from the DATA= data set to be included in the OUT= data set. If there is no ID statement, all variables from the DATA= data set are included in the OUT= data set. The ID variables can be character or numeric.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the variables to be used in computing scores. These variables must be in both the DATA= and SCORE= input data sets and must be numeric. If you do not specify a VAR statement, the procedure uses all numeric variables in the SCORE= data set. You should almost always specify a VAR statement with PROC SCORE because you would rarely use all the numeric variables in your data set to compute scores.

Details: SCORE Procedure

Missing Values

If one of the scoring variables in the DATA= data set has a missing value for an observation, all the scores have missing values for that observation. The exception to this criterion is that if the PREDICT option is specified, the variable with a coefficient of –1 can tolerate a missing value and still produce a prediction score. Also, a variable with a coefficient of 0 can tolerate a missing value.

If a scoring coefficient in the SCORE= data set has a missing value for an observation, the coefficient is not used in creating the new score variable for the observation. In other words, missing values of scoring coefficients are treated as zeros. This treatment affects only the observation in which the missing value occurs.

Regression Parameter Estimates from PROC REG

If the SCORE= data set is an OUTEST= data set produced by PROC REG and if you specify TYPE=PARMS, the interpretation of the new score variables depends on the PROC SCORE options chosen and the variables listed in the VAR statement. If the VAR statement contains only the independent variables used in a model in PROC REG, the new score variables give the predicted values. If the VAR statement contains the dependent variables and the independent variables used in a model in PROC REG, the interpretation of the new score variables depends on the PROC SCORE options chosen. If you omit both the PREDICT and the RESIDUAL options, the new score variables give negative residuals (PREDICT–ACTUAL). If you specify the RESIDUAL option, the new score variables give positive residuals (ACTUAL–PREDICT). If you specify the PREDICT option, the new score variables give predicted values.

Unless you specify the NOINT option for PROC REG, the OUTEST= data set contains the variable Intercept. The SCORE procedure uses the intercept value in computing the scores.

Output Data Set

PROC SCORE produces an output data set but displays no output. The output OUT= data set contains the following variables:

- the ID variables, if any
- all variables from the DATA= data set, if no ID variables are specified
- the BY variables, if any
- the new score variables, named from the _NAME_ or _MODEL_ values in the SCORE= data set

Computational Resources

Let

v = number of variables used in computing scores
s = number of new score variables
b = maximum number of new score variables in a BY group
n = original input value

Memory

The array storage required is approximately $8(4v + (3 + v)b + s)$ bytes. When you do not use BY processing, the array storage required is approximately $8(4v + (4 + v)s)$ bytes.

Time

The time required to construct the scoring matrix is roughly proportional to vs, and the time needed to compute the scores is roughly proportional to nvs.

Examples: SCORE Procedure

The following three examples use a subset of the Fitness data set. The complete data set is given in Chapter 85, "The REG Procedure."

Example 88.1: Factor Scoring Coefficients

This example shows how to use PROC SCORE with factor scoring coefficients. First, the FACTOR procedure produces an output data set containing scoring coefficients in observations identified by _TYPE_='SCORE'. These data, together with the original data set Fitness, are supplied to PROC SCORE, resulting in a data set containing scores Factor1 and Factor2. The following statements produce Output 88.1.1 through Output 88.1.3:

```
   /* This data set contains only the first 12 observations   */
   /* from the full data set used in the chapter on PROC REG. */
data Fitness;
   input Age Weight Oxygen RunTime RestPulse RunPulse @@;
   datalines;
44 89.47  44.609 11.37 62 178     40 75.07  45.313 10.07 62 185
44 85.84  54.297  8.65 45 156     42 68.15  59.571  8.17 40 166
38 89.02  49.874  9.22 55 178     47 77.45  44.811 11.63 58 176
40 75.98  45.681 11.95 70 176     43 81.19  49.091 10.85 64 162
44 81.42  39.442 13.08 63 174     38 81.87  60.055  8.63 48 170
44 73.03  50.541 10.13 45 168     45 87.66  37.388 14.03 56 186
;

proc factor data=Fitness outstat=FactOut
            method=prin rotate=varimax score;
   var Age Weight RunTime RunPulse RestPulse;
   title 'Factor Scoring Example';
run;

proc print data=FactOut;
   title2 'Data Set from PROC FACTOR';
run;

proc score data=Fitness score=FactOut out=FScore;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=FScore;
   title2 'Data Set from PROC SCORE';
run;
```

Output 88.1.1 shows the PROC FACTOR output. The scoring coefficients for the two factors are shown at the end of the PROC FACTOR output.

Output 88.1.1 Creating an OUTSTAT= Data Set with PROC FACTOR

Factor Scoring Example

The FACTOR Procedure

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

Factor Scoring Example

The FACTOR Procedure
Initial Factor Method: Principal Components

Prior Communality Estimates: ONE

Eigenvalues of the Correlation Matrix: Total = 5 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	2.30930638	1.11710686	0.4619	0.4619
2	1.19219952	0.30997249	0.2384	0.7003
3	0.88222702	0.37965990	0.1764	0.8767
4	0.50256713	0.38886717	0.1005	0.9773
5	0.11369996		0.0227	1.0000

2 factors will be retained by the MINEIGEN criterion.

Factor Pattern

	Factor1	Factor2
Age	0.29795	0.93675
Weight	0.43282	-0.17750
RunTime	0.91983	0.28782
RunPulse	0.72671	-0.38191
RestPulse	0.81179	-0.23344

Variance Explained by Each Factor

Factor1	Factor2
2.3093064	1.1921995

Final Communality Estimates: Total = 3.501506

Age	Weight	RunTime	RunPulse	RestPulse
0.96628351	0.21883401	0.92893333	0.67396207	0.71349297

Output 88.1.1 *continued*

Factor Scoring Example

The FACTOR Procedure
Rotation Method: Varimax

Orthogonal Transformation Matrix

	1	2
1	0.92536	0.37908
2	-0.37908	0.92536

Rotated Factor Pattern

	Factor1	Factor2
Age	-0.07939	0.97979
Weight	0.46780	-0.00018
RunTime	0.74207	0.61503
RunPulse	0.81725	-0.07792
RestPulse	0.83969	0.09172

Variance Explained by Each Factor

Factor1	Factor2
2.1487753	1.3527306

Final Communality Estimates: Total = 3.501506

Age	Weight	RunTime	RunPulse	RestPulse
0.96628351	0.21883401	0.92893333	0.67396207	0.71349297

Factor Scoring Example

The FACTOR Procedure
Rotation Method: Varimax

Scoring Coefficients Estimated by Regression

Squared Multiple Correlations of the Variables with Each Factor

Factor1	Factor2
1.0000000	1.0000000

Standardized Scoring Coefficients

	Factor1	Factor2
Age	-0.17846	0.77600
Weight	0.22987	-0.06672
RunTime	0.27707	0.37440
RunPulse	0.41263	-0.17714
RestPulse	0.39952	-0.04793

Output 88.1.2 lists the OUTSTAT= data set from PROC FACTOR. Note that observations 18 and 19 have _TYPE_='SCORE'. Observations 1 and 2 have _TYPE_='MEAN' and _TYPE_='STD', respectively. These four observations are used by PROC SCORE.

Output 88.1.2 OUTSTAT= Data Set from PROC FACTOR Reproduced with PROC PRINT

Factor Scoring Example
Data Set from PROC FACTOR

Obs	_TYPE_	_NAME_	Age	Weight	RunTime	RunPulse	RestPulse
1	MEAN		42.4167	80.5125	10.6483	172.917	55.6667
2	STD		2.8431	6.7660	1.8444	8.918	9.2769
3	N		12.0000	12.0000	12.0000	12.000	12.0000
4	CORR	Age	1.0000	0.0128	0.5005	-0.095	-0.0080
5	CORR	Weight	0.0128	1.0000	0.2637	0.173	0.2396
6	CORR	RunTime	0.5005	0.2637	1.0000	0.556	0.6620
7	CORR	RunPulse	-0.0953	0.1731	0.5555	1.000	0.4853
8	CORR	RestPulse	-0.0080	0.2396	0.6620	0.485	1.0000
9	COMMUNAL		0.9663	0.2188	0.9289	0.674	0.7135
10	PRIORS		1.0000	1.0000	1.0000	1.000	1.0000
11	EIGENVAL		2.3093	1.1922	0.8822	0.503	0.1137
12	UNROTATE	Factor1	0.2980	0.4328	0.9198	0.727	0.8118
13	UNROTATE	Factor2	0.9368	-0.1775	0.2878	-0.382	-0.2334
14	TRANSFOR	Factor1	0.9254	-0.3791	.	.	.
15	TRANSFOR	Factor2	0.3791	0.9254	.	.	.
16	PATTERN	Factor1	-0.0794	0.4678	0.7421	0.817	0.8397
17	PATTERN	Factor2	0.9798	-0.0002	0.6150	-0.078	0.0917
18	SCORE	Factor1	-0.1785	0.2299	0.2771	0.413	0.3995
19	SCORE	Factor2	0.7760	-0.0667	0.3744	-0.177	-0.0479

Since the PROC SCORE statement does not contain the NOSTD option, the data in the Fitness data set are standardized before scoring. For each variable specified in the VAR statement, the mean and standard deviation are obtained from the FactOut data set. For each observation in the Fitness data set, the variables are then standardized. For example, for observation 1 in the Fitness data set, the variable Age is standardized to $0.5569 = [(44 - 42.4167)/2.8431]$.

After the data in the Fitness data set are standardized, the standardized values of the variables in the VAR statement are multiplied by the matching coefficients in the FactOut data set, and the resulting products are summed. This sum is output as a value of the new score variable.

Output 88.1.3 displays the FScore data set produced by PROC SCORE. This data set contains the variables Age, Weight, Oxygen, RunTime, RestPulse, and RunPulse from the Fitness data set. It also contains Factor1 and Factor2, the two new score variables.

Output 88.1.3 OUT= Data Set from PROC SCORE Reproduced with PROC PRINT

Factor Scoring Example
Data Set from PROC SCORE

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	Factor1	Factor2
1	44	89.47	44.609	11.37	62	178	0.82129	0.35663
2	40	75.07	45.313	10.07	62	185	0.71173	-0.99605
3	44	85.84	54.297	8.65	45	156	-1.46064	0.36508
4	42	68.15	59.571	8.17	40	166	-1.76087	-0.27657
5	38	89.02	49.874	9.22	55	178	0.55819	-1.67684
6	47	77.45	44.811	11.63	58	176	-0.00113	1.40715
7	40	75.98	45.681	11.95	70	176	0.95318	-0.48598
8	43	81.19	49.091	10.85	64	162	-0.12951	0.36724
9	44	81.42	39.442	13.08	63	174	0.66267	0.85740
10	38	81.87	60.055	8.63	48	170	-0.44496	-1.53103
11	44	73.03	50.541	10.13	45	168	-1.11832	0.55349
12	45	87.66	37.388	14.03	56	186	1.20836	1.05948

Example 88.2: Regression Parameter Estimates

In this example, PROC REG computes regression parameter estimates for the Fitness data. (See Example 88.1 to for more information about how to create the Fitness data set.) The parameter estimates are output to a data set and used as scoring coefficients. For the first part of this example, PROC SCORE is used to score the Fitness data, which are the same data used in the regression.

In the second part of this example, PROC SCORE is used to score a new data set, Fitness2. For PROC SCORE, the TYPE= specification is PARMS, and the names of the score variables are found in the variable _MODEL_, which gets its values from the model label. The following code produces Output 88.2.1 through Output 88.2.3:

```
proc reg data=Fitness outest=RegOut;
   OxyHat: model Oxygen=Age Weight RunTime RunPulse RestPulse;
   title 'Regression Scoring Example';
run;

proc print data=RegOut;
   title2 'OUTEST= Data Set from PROC REG';
run;

proc score data=Fitness score=RegOut out=RScoreP type=parms;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=RScoreP;
   title2 'Predicted Scores for Regression';
run;

proc score data=Fitness score=RegOut out=RScoreR type=parms;
   var Oxygen Age Weight RunTime RunPulse RestPulse;
run;
```

```
proc print data=RScoreR;
   title2 'Negative Residual Scores for Regression';
run;
```

Output 88.2.1 shows the PROC REG output. The column labeled “Parameter Estimates” lists the parameter estimates. These estimates are output to the RegOut data set.

Output 88.2.1 Creating an OUTEST= Data Set with PROC REG

Regression Scoring Example

The REG Procedure
Model: OxyHat
Dependent Variable: Oxygen

Number of Observations Read	12
Number of Observations Used	12

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	509.62201	101.92440	15.80	0.0021
Error	6	38.70060	6.45010		
Corrected Total	11	548.32261			

Root MSE	2.53970	**R-Square**	0.9294
Dependent Mean	48.38942	**Adj R-Sq**	0.8706
Coeff Var	5.24847		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	151.91550	31.04738	4.89	0.0027
Age	1	-0.63045	0.42503	-1.48	0.1885
Weight	1	-0.10586	0.11869	-0.89	0.4068
RunTime	1	-1.75698	0.93844	-1.87	0.1103
RunPulse	1	-0.22891	0.12169	-1.88	0.1090
RestPulse	1	-0.17910	0.13005	-1.38	0.2176

Output 88.2.2 lists the RegOut data set. Note that _TYPE_='PARMS' and _MODEL_='OXYHAT', which are from the label in the MODEL statement in PROC REG.

Output 88.2.2 OUTEST= Data Set from PROC REG Reproduced with PROC PRINT

Regression Scoring Example
OUTEST= Data Set from PROC REG

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	Age	Weight	RunTime	RunPulse	RestPulse	Oxygen
1	OxyHat	PARMS	Oxygen	2.53970	151.916	-0.63045	-0.10586	-1.75698	-0.22891	-0.17910	-1

Output 88.2.3 lists the data sets created by PROC SCORE. Since the SCORE= data set does not contain observations with _TYPE_='MEAN' or _TYPE_='STD', the data in the Fitness data set are not standardized before scoring. The SCORE= data set contains the variable Intercept, so this intercept value is used in computing the score. To produce the RScoreP data set, the VAR statement in PROC SCORE includes only the independent variables from the model in PROC REG. As a result, the OxyHat variable contains predicted values. To produce the RScoreR data set, the VAR statement in PROC SCORE includes both the dependent variables and the independent variables from the model in PROC REG. As a result, the OxyHat variable contains negative residuals (PREDICT–ACTUAL) as shown in Output 88.2.4. If the RESIDUAL option is specified, the variable OxyHat contains positive residuals (ACTUAL–PREDICT). If the PREDICT option is specified, the OxyHat variable contains predicted values.

Output 88.2.3 Predicted Scores from the OUT= Data Set Created by PROC SCORE

Regression Scoring Example
Predicted Scores for Regression

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	44	89.47	44.609	11.37	62	178	42.8771
2	40	75.07	45.313	10.07	62	185	47.6050
3	44	85.84	54.297	8.65	45	156	56.1211
4	42	68.15	59.571	8.17	40	166	58.7044
5	38	89.02	49.874	9.22	55	178	51.7386
6	47	77.45	44.811	11.63	58	176	42.9756
7	40	75.98	45.681	11.95	70	176	44.8329
8	43	81.19	49.091	10.85	64	162	48.6020
9	44	81.42	39.442	13.08	63	174	41.4613
10	38	81.87	60.055	8.63	48	170	56.6171
11	44	73.03	50.541	10.13	45	168	52.1299
12	45	87.66	37.388	14.03	56	186	37.0080

Output 88.2.4 Residual Scores from the OUT= Data Set Created by PROC SCORE

Regression Scoring Example
Negative Residual Scores for Regression

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	44	89.47	44.609	11.37	62	178	-1.73195
2	40	75.07	45.313	10.07	62	185	2.29197
3	44	85.84	54.297	8.65	45	156	1.82407
4	42	68.15	59.571	8.17	40	166	-0.86657
5	38	89.02	49.874	9.22	55	178	1.86460
6	47	77.45	44.811	11.63	58	176	-1.83542
7	40	75.98	45.681	11.95	70	176	-0.84811
8	43	81.19	49.091	10.85	64	162	-0.48897
9	44	81.42	39.442	13.08	63	174	2.01935
10	38	81.87	60.055	8.63	48	170	-3.43787
11	44	73.03	50.541	10.13	45	168	1.58892
12	45	87.66	37.388	14.03	56	186	-0.38002

The second part of this example uses the parameter estimates to score a new data set. The following statements produce Output 88.2.5 and Output 88.2.6:

```
   /* The FITNESS2 data set contains observations 13-16 from */
   /* the FITNESS data set used in EXAMPLE 2 in the PROC REG */
   /* chapter.                                                */
data Fitness2;
   input Age Weight Oxygen RunTime RestPulse RunPulse;
   datalines;
45  66.45  44.754  11.12  51  176
47  79.15  47.273  10.60  47  162
54  83.12  51.855  10.33  50  166
49  81.42  49.156   8.95  44  180
;

proc print data=Fitness2;
   title 'Regression Scoring Example';
   title2 'New Raw Data Set to be Scored';
run;

proc score data=Fitness2 score=RegOut out=NewPred type=parms
           nostd predict;
   var Oxygen Age Weight RunTime RunPulse RestPulse;
run;

proc print data=NewPred;
   title2 'Predicted Scores for Regression';
   title3 'for Additional Data from FITNESS2';
run;
```

Output 88.2.5 lists the Fitness2 data set.

Output 88.2.5 Listing of the Fitness2 Data Set

Regression Scoring Example
New Raw Data Set to be Scored

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse
1	45	66.45	44.754	11.12	51	176
2	47	79.15	47.273	10.60	47	162
3	54	83.12	51.855	10.33	50	166
4	49	81.42	49.156	8.95	44	180

PROC SCORE scores the Fitness2 data set by using the parameter estimates in the RegOut data set. These parameter estimates result from fitting a regression equation to the Fitness data set. The NOSTD option is specified, so the raw data are not standardized before scoring. (However, the NOSTD option is not necessary here. The SCORE= data set does not contain observations with _TYPE_='MEAN' or _TYPE_='STD', so standardization is not performed.) The VAR statement contains the dependent variables and the independent variables used in PROC REG. In addition, the PREDICT option is specified. This combination gives predicted values for the new score variable. The name of the new score variable is OxyHat, from the value of the _MODEL_ variable in the SCORE= data set. Output 88.2.6 shows the data set produced by PROC SCORE.

Output 88.2.6 Predicted Scores from the OUT= Data Set Created by PROC SCORE and Reproduced Using PROC PRINT

Regression Scoring Example
Predicted Scores for Regression
for Additional Data from FITNESS2

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	45	66.45	44.754	11.12	51	176	47.5507
2	47	79.15	47.273	10.60	47	162	49.7802
3	54	83.12	51.855	10.33	50	166	43.9682
4	49	81.42	49.156	8.95	44	180	47.5949

Example 88.3: Custom Scoring Coefficients

This example uses a specially created custom scoring data set and produces Output 88.3.1 and Output 88.3.2. The first scoring coefficient creates a variable that is Age–Weight; the second scoring coefficient evaluates the variable RunPulse–RstPulse; and the third scoring coefficient totals all six variables. Since the scoring coefficients data set (data set A) does not contain any observations with _TYPE_='MEAN' or _TYPE_='STD', the data in the Fitness data set (see Example 88.1) are not standardized before scoring.

The following statements produce Output 88.3.1 and Output 88.3.2:

```
data A;
   input _type_ $ _name_ $
         Age Weight RunTime RunPulse RestPulse;
   datalines;
SCORE   AGE_WGT   1 -1  0  0  0
SCORE   RUN_RST   0  0  0  1 -1
SCORE   TOTAL     1  1  1  1  1
;

proc print data=A;
   title 'Constructed Scoring Example';
   title2 'Scoring Coefficients';
run;

proc score data=Fitness score=A out=B;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=B;
   title2 'Scored Data';
run;
```

Output 88.3.1 Custom Scoring Data Set and Scored Fitness Data: PROC PRINT

Constructed Scoring Example
Scoring Coefficients

Obs	_type_	_name_	Age	Weight	RunTime	RunPulse	RestPulse
1	SCORE	AGE_WGT	1	-1	0	0	0
2	SCORE	RUN_RST	0	0	0	1	-1
3	SCORE	TOTAL	1	1	1	1	1

Output 88.3.2 Custom Scored Fitness Data: PROC PRINT

Constructed Scoring Example
Scored Data

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	AGE_WGT	RUN_RST	TOTAL
1	44	89.47	44.609	11.37	62	178	-45.47	116	384.84
2	40	75.07	45.313	10.07	62	185	-35.07	123	372.14
3	44	85.84	54.297	8.65	45	156	-41.84	111	339.49
4	42	68.15	59.571	8.17	40	166	-26.15	126	324.32
5	38	89.02	49.874	9.22	55	178	-51.02	123	369.24
6	47	77.45	44.811	11.63	58	176	-30.45	118	370.08
7	40	75.98	45.681	11.95	70	176	-35.98	106	373.93
8	43	81.19	49.091	10.85	64	162	-38.19	98	361.04
9	44	81.42	39.442	13.08	63	174	-37.42	111	375.50
10	38	81.87	60.055	8.63	48	170	-43.87	122	346.50
11	44	73.03	50.541	10.13	45	168	-29.03	123	340.16
12	45	87.66	37.388	14.03	56	186	-42.66	130	388.69

References

Chaseling, J. (1996), “Standard Test Results of Students at Three Types of Schools,” Sample data, Faculty of Environmental Sciences, Griffith University, Queensland, Australia.

Subject Index

Syntax Index