

5 Merging, data set options, GLM

1. More on SET and MERGE
2. Automatic system variables
3. Merging with duplicates
4. Finding unmatched observations in a merge
5. Finding discrepant values for the same observation
6. Data set options
7. Data step programming: IF-THEN-ELSE
8. Proc GLM: analysis of variance

1

More on SET and MERGE (*LSB Chapter 6*)

With merge BY *variable* and set BY *variable*, two secret variables are created (*LSB §6.14*) :

`first.variable` = 1 for the first observation with this value of variable, = 0 otherwise

`last.variable` = 1 for the last observation with this value of variable, = 0 otherwise

These can be very useful when the data has multiple observations for each ID.

Another secret variable is: `_N_` = the number of times SAS has gone through the whole data step

These variables are not saved, and only exist during the data step that creates them. To keep them, you must make a new variable equal to the one you want to keep.

2

Merging with missing and duplicate values

```
data D;
  input id mass pH;
  cards;
  13 11267 7.8
  14 3568 8.2
  15 4103 5.1
  15 4102 5.1
  . 2321 5.6
;
data E;
  input id mass pH;
  cards;
  13 11267 7.8
  14 3568 8.2
  15 4103 5.1
  . 2321 5.6
  . 4681 7.8
;
proc sort data=D; by id;
proc sort data=E; by id;

data M;
  merge D E;  by id;
```

3

Obs	id	mass	pH
1	.	2321	5.6
2	.	4681	7.8
3	13	11267	7.8
4	14	3568	8.2
5	15	4103	5.1
6	15	4102	5.1

Missing values are sorted as $-\infty$, so they come first.

What would happen if the order of the duplicates was reversed?

4

```

data d1;
  input id mass pH;
  cards;
  13 11267 7.8
  14 3568 8.2
  15 4102 5.1  switched
  15 4103 5.1
  . 2321 5.6
  ;
data e1;
  input id mass pH;
  cards;
  13 11267 7.8
  14 3568 8.2
  15 4103 5.1
  . 4681 7.8  switched
  . 2321 5.6
  ;
proc sort data=d1; by id;
proc sort data=e1; by id;

data m;
  merge d1 e1;  by id;

```

5

Obs	id	mass	pH
1	.	4681	7.8
2	.	2321	5.6
3	13	11267	7.8
4	14	3568	8.2
5	15	4103	5.1
6	15	4103	5.1

One of the duplicates was overwritten.

Don't merge by one variable that has missing values or duplicates.

6

When there are duplicates, sort by 2 variables and merge by both:

```
proc sort data=d1; by id mass ;
proc sort data=e1; by id mass ;

data m2;
  merge d1 e1;
  by id mass ;
```

Obs	id	mass	pH
1	.	2321	5.6
2	.	4681	7.8
3	13	11267	7.8
4	14	3568	8.2
5	15	4102	5.1
6	15	4103	5.1

7

Finding unmatched observations in a MERGE

An “unmatched observation” appears in only one of the data sets being merged:

Data A			Data B		
id	color	mass	id	mass	pH
12	orange	3650	13	11267	7.8
13	blue	3877	14	3568	8.2
15	yellow	4103	15	4103	5.1

We would like to merge A with B by `id`, and then get a list of the unmatched observations that tells which data set they came from.

We use a dataset option (`IN = variable`) to create an indicator that labels the source.

```
Proc Sort data = A; by id; first SORT by ID
```

```
Proc Sort data = B; by id;
```

```
data G;
```

```
merge A(in = in_a) B(in = in_b) ;
```

```
by id;
```

```
from_a = in_a ; save these variables by renaming them
```

```
from_b = in_b ;
```

```
Proc Print;
```

The whole merged data set:

Obs	id	color	mass	pH	from_a	from_b
1	12	orange	3650	.	1	0
2	13	blue	11267	7.8	1	1
3	14		3568	8.2	0	1
4	15	yellow	4103	5.1	1	1

9

To get a list of the unmatched observations:

```
data H;
```

```
set G;
```

```
if (from_a NE from_b);
```

```
Proc Print;
```

Obs	id	color	mass	pH	from_a	from_b
1	12	orange	3650	.	1	0
2	14		3568	8.2	0	1

How could we get this list with all the A observations first?

Finding discrepant values for the same observation in a MERGE

Data A			Data B		
id	color	mass	id	mass	pH
12	orange	3650	13	11267	7.8
13	blue	3877	14	3568	8.2
15	yellow	4103	15	4103	5.1

We would like to merge A with B, and then get a list of the observations that have different values for `mass`.

What happens with this?

```
Proc Sort data = A; by id;
Proc Sort data = B; by id;
data I;
  merge A B ;
  by id;
  if (mass NE mass);
```

11

```
Proc Sort data = A; by id;
Proc Sort data = B; by id;
```

```
data I;
  merge A(rename = (mass = mass_a) ) B(rename = (mass = mass_b) ) ;
  by id;
Proc Print;
```

Obs	id	color	mass_a	mass_b	pH
1	12	orange	3650	.	.
2	13	blue	3877	11267	7.8
3	14		.	3568	8.2
4	15	yellow	4103	4103	5.1

12

List of observations with discrepant values:

```
data J;  
  set I;  
  if (mass_a NE mass_b) ;  
Proc Print;
```

Obs	id	color	mass_a	mass_b	pH
1	12	orange	3650	.	.
2	13	blue	3877	11267	7.8
3	14		.	3568	8.2

13

Data Set Options (LSB §6.9)

Data set options change the way SAS reads a data set into a procedure or into a data step.

The option(s) go in parentheses after the data set name: `DataName(option)`

`FIRSTOBS` = tells SAS where to start processing the data

`OBS` = tells SAS where to stop processing the data

So we need `FIRSTOBS < OBS` to read any observations

These options work on observations (rows) of the data set.

14

These options work on variables (columns) of the data set:

DROP = *variable list*: specifies *variables* to be excluded

KEEP = *variable list*: specifies *variables* to be included

RENAME = (*oldname1=newname1 oldname2=newname2*) changes the names of variables *oldname1* and *oldname2*

IN = *new variable*: in a SET or MERGE creates a variable that indicates whether this data set contributed to the current observation.

This variable only exists during the datastep: if you print the data it won't appear.

To keep it, create a new variable equal to the IN-variable.

15

Data step programming: IF-THEN-ELSE (LSB §3.4–3.5)

We have already used the *subsetting IF*:

```
IF (condition);
```

Observation is included in the data set only if (condition) is true for that observation.

Second part of IF is THEN

```
IF (condition) THEN statement;  
IF (educ_years GE 12) THEN education = "high school graduate";
```

statement is executed only if (condition) is true for that observation.

16

For more than one conditional statements, use DO; . . . ; END;

```
IF (id = 1207645) THEN DO;  
  age = 46;  
  BMI = 32.4;  
  clinic = "Fairview";  
END;
```

statements between DO and END are executed only if (condition) is true for that observation.

SAS will write an error message in the log file for a DO-loop without an END.

17

IF - THEN - ELSE

Can also specify statements for both results of testing a condition:

```
IF (condition) THEN statement_A;  
  ELSE statement_B;  
  
IF (educ_years GE 12) THEN education = "high school graduate";  
  ELSE education = "dropped out";
```

Either THEN or ELSE can start a DO-loop.

18

```
IF (average GE 90.0) THEN grade = "A";
  ELSE IF (average GE 80.0) THEN grade = "B";
  ELSE IF (average GE 70.0) THEN grade = "C";
  ELSE IF (0 LE average < 70.0 ) THEN grade = "F";
  ELSE grade = "?";
```

What happens to observations with average missing?

19

GLM: Analysis of Variance (ANOVA)

SAS began as software to compute regression and ANOVA. SAS now has several procedures that compute ANOVA by ordinary least squares,

ANOVA, GLM, Reg, MIXED, GENMOD

Don't ever use ANOVA: it only handles balanced designs and will not produce residuals for model checking.

We will start with GLM, an acronym for *General Linear Model*.

Example: Multicenter Depression Trial (source: Dmitrienko *et. al.* (2005))

This clinical trial compared an experimental drug (D) with placebo (P) in patients with major depression. The primary efficacy measure was change in Hamilton depression rating scale from baseline to the end of the 9-week treatment.

Randomization was stratified within centers (numbered 1–5).

20

```
Proc Print data = pubh.hamd (obs=6) ; print only the first 6 observations
```

Obs	drug	change	center
1	P	18	1
2	P	14	1
3	D	23	1
4	D	18	1
5	P	10	1
6	P	17	1

The study aim was to compare experimental drug (D) to placebo (P) , but because randomization was stratified by center, we must take center (stratum) into account. **Analysis must always match study design.**

Center is a stratum or block, so this is a randomized block design.

We will model the response (change in Hamilton depression scale) on two factors: center and treatment, plus a possible interaction between treatment and center.

21

Two-factor ANOVA model

Let x_{ijk} = observation from subject k assigned to drug i at center j . One way to write our model:

$$x_{ijk} = \mu + \tau_i + c_j + (\tau c)_{ij} + \varepsilon_{ijk},$$

with errors ε_{ijk} that are independent Normal variables with zero mean and standard deviation σ .

$\{c_j\}$ are the center differences from the overall mean μ .

$\{\tau_i\}$ are the treatment differences from the overall mean μ .

$\{(\tau c)_{ij}\}$ are the interaction terms: treatment differences from center to center.

To solve this, we add constraints $0 = \sum_i \tau_i = \sum_j c_j = \sum_i (\tau c)_{ij} = \sum_j (\tau c)_{ij}$.

22

Our explanatory variables are both categorical factors, or **CLASS variables**.

```
Proc GLM;
```

```
  CLASS A B;
```

```
  MODEL response(s) = A B A*B;
```

```
  MEANS main effects or interactions; group means and within-group SDs
```

```
  LSMEANS main effects or interactions / pdiff stderr;
```

```
    compare means with pooled SEs
```

```
Proc GLM data = pubh.hamd;
```

```
  class drug center;
```

```
  model change = center drug center*drug;
```

```
  means center*drug;
```

```
  lsmeans center*drug / pdiff stderr;
```

23

The GLM Procedure

Class Level Information

Class	Levels	Values	<i>Missing value codes?</i>
drug	2	D P	
center	5	1 2 3 4 5	

Number of Observations	Read	100
------------------------	------	-----

Number of Observations	Used	100
------------------------	------	-----

Observations with missing values for response or any predictors are not used.

How many observations dropped for missing values?

24

Dependent Variable: change

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	9	1482.624997	164.736111	7.43	<.0001
Error	90	1994.375003	22.159722		
Corrected Total	99	3477.000000			

R-Square	Coeff Var	Root MSE	change Mean
0.426409	29.60636	4.707411	15.90000

Source	DF	Type I SS	Mean Square	F Value	Pr > F
center	4	85.4035521	21.3508880	0.96	0.4316
drug	1	889.7756912	889.7756912	40.15	<.0001
drug*center	4	507.4457539	126.8614385	5.72	0.0004

Source	DF	Type III SS	Mean Square	F Value	Pr > F
center	4	91.4580063	22.8645016	1.03	0.3953
drug	1	709.8195519	709.8195519	32.03	<.0001
drug*center	4	507.4457539	126.8614385	5.72	0.0004

25

R-square = $SS(\text{Model}) / SS(\text{Total})$, the proportion of total variability explained by the model

Root MSE = Root Mean Square Error, $\sqrt{MS(\text{Error})} = \hat{\sigma} = \text{pooled SD}$

change Mean = is the mean response from all observations, the “grand mean.”

Coeff Var = $100\% * (\text{Root MSE}) / (\text{grand mean})$

A *cell* is a subgroup of observations defined by the intersection of single levels of each model term. Depression study, with 5 clinics and 2 treatments, has 10 cells.

Type I SS also called **sequential sums of squares**, are the incremental improvement in error sums of squares as each effect is added to the model. They can be computed by fitting the model in steps and recording the difference in error sum of squares at each step. With unbalanced data, Type I hypotheses are *functions of the cell counts*: stratum means are compared weighted by the number in the stratum. The hypothesis of no treatment effect depends on the number of patients at each clinic.

26

Type III SS can be regarded, in simple situations, as the Type I SS for each term if it were the final term in the model. Type III SS are invariant to changes in the order of terms in the model, and do not depend on cell counts. Stratum means are compared directly without weights.

A two-factor ANOVA where all cells have the same size is called *balanced* and in this case, Type I SS = Type III SS.

References:

§1.2.1 of Dmitrienko, Molenberghs, Chuang-Stein, Offen (2005) *Analysis of Clinical Trials Using SAS: A Practical Guide*

“Hypothesis Testing in Proc GLM,” in the SAS documentation for GLM (details)

Ch 10–11 of Yandell (1997) *Practical Data Analysis of Designed Experiments*