BIOMETRICS INFORMATION

(You're 95% likely to need this information)

PAMPHLET NO. # 47 DATE: April 22, 1994

SUBJECT: SAS: Adding Observations when Class Variables (e.g. species list) are Missing

Occasionally a dataset does not contain an observation for every combination of values for the class variables. A common example occurs when lists of species are made for each plot in a study but missing species are not listed. In subsequent analysis, it may be desirable to add an observation for each of these missing species and give them zero cover values. The following programs provide some examples of how to add these observations.

The example dataset has three class variables, `plot` for a categorical variable, `sp` for a variable containing species names, and `spnum` using numbers to code for species. One or more combinations of `plot` and `spnum/sp` do not have any observations (records) but the response variable cov (for percent cover) is be set to 0 (zero) for each of these missing records (code is another response variable). The following dataset will be our example.

```
data begin;                          /* create a new dataset called begin        */
input plot spnum sp $ cov code $; /* input statement using list format        */
cards;                               /* data begins on next line                 */
1 1 A 21 X
1 3 C 44 W
1 4 D 53 Z
2 2 B 26 X
2 4 D 33 K
3 1 A 20 J
3 2 B 25 W
3 5 E 42 X
;                                    /* end of data                              */
proc sort; by plot spnum;            /* sort dataset by plot and sp              */
```

A simple method is to create a second dataset[1] containing all the combinations of `plot` and `spnum`. This is then combined with the original dataset as in the following program.

```
data allcat;                    /* create a new dataset called allcat          */
  do plot = 1 to 3;             /* create plot values                          */
     do spnum = 1 to 5;         /* create spnum value (must know range)        */
     output;  end; end;         /* output obs and end both do loops            */
run;
data both;                      /* create a new dataset called both            */
  merge allcat begin;           /* merge previous datasets allcat and begin    */
  by plot spnum;                /* match obs by plot and spnum values          */
  if cov = . then cov = 0;      /* if cov is missing the make the value zero   */
run;  proc print; run;          /* run and print the data                      */
```

[1] Note that it is best if this second dataset contains **only** the class variables.

The output from this program is:

OBS	PLOT	SPNUM	SP	COV	CODE
1	1	1	A	21	X
2	1	2		0	
3	1	3	C	44	W
4	1	4	D	53	Z
5	1	5		0	
6	2	1		0	
7	2	2	B	26	X
8	2	3		0	
9	2	4	D	33	K
10	2	5		0	
11	3	1	A	20	J
12	3	2	B	25	W
13	3	3		0	
14	3	4		0	
15	3	5	E	42	X

Notice that the added observations have a zero for `cov` and a missing value (blank) for `code`.

To illustrate this problem with a categorical variable, suppose that the class variables were `plot` and `sp` instead of `plot` and `spnum`. The following program shows how to create the missing observations in this situation.

```
proc format;                    /* create a species list assigning numbers to species */
  value splst  1 ='A'  2='B'    3 ='C'  4 ='D'  5 ='E';
run;
proc sort; by plot sp;          /* sort by plot and sp variables                */
data allcat;                    /* create a new dataset called allcat           */
  do plot = 1 to 3;             /* create plot values                           */
     do spnum = 1 to 5;         /* create spnum value (must know values)        */
        sp = put(spnum,splst.);                 /* create sp variable          */
        drop spnum;
     output;  end; end;         /* output obs and end both do loops             */
run;
data both;                      /* create a new dataset called both             */
  merge allcat begin;           /* merge previous datasets allcat and begin     */
  by plot sp;                   /* match obs by plot and sp values              */
  if cov = . then cov = 0;      /* if cov is missing the make the value zero    */
run;  proc print; run;          /* run and print the data                       */
```

Notice that the `put` function is used to convert the numerical values of `spnum` into the categorical values of `sp`. The `put` function assigns `sp` values according to the current value of `spnum` and the `splst` format. For example, when `spnum` is 2, the `put` function goes to the `splst` format table to determine which label is assigned to 2 and assigns that label as a value to the variable `sp` (for this example, this value is B).

The output from this program is:

OBS	PLOT	SP	SPNUM	COV	CODE
1	1	A	1	21	X
2	1	B	.	0	
3	1	C	3	44	W
4	1	D	4	53	Z
5	1	E	.	0	
6	2	A	.	0	
7	2	B	2	26	X
8	2	C	.	0	
9	2	D	4	33	K
10	2	E	.	0	
11	3	A	1	20	J
12	3	B	2	25	W
13	3	C	.	0	
14	3	D	.	0	
15	3	E	5	42	X

Again, `cov` has zero values for the added observations and `code` has missing values (blanks).

Obtaining the `splst` format list was easy for this example since there were only 5 short names. But since species lists usually contain many species, each having a long label or name, creating this table would be a lot of work. The following program can be used to obtain a complete species list for the dataset, even when the total number of species is unknown. Then it is simple to `include` the file into your program.

```
proc sort data=begin; by sp; run; /* sort dataset by sp                        */
data _null_;                      /* no dataset will be created               */
  set begin;                      /* use the begin dataset                    */
  by sp;                          /* work by values of the sp variable        */
  file 'sp.lst';                  /* create a file called sp.lst              */
  if last.sp then do;             /* if last obs in sp group then do:         */
    spn+1;                        /* species number counter                   */
    put @17 spn ' = ' "'"  sp  "'"  ;  /* add one line to file for each sp    */
  end;                            /* end of do loop                           */
run;                              /* run data step                            */
```

The output in the file called `sp.lst` is:

```
1  = 'A '
2  = 'B '
3  = 'C '
4  = 'D '
5  = 'E '
```

Note that the species list is sorted alphabetically.

The above program is 'bare bones' and provides the basic file necessary to create the `proc format` statement. The following program is more elegant but not often worth the trouble. It also creates a file called `sp.lst` but which now contains the full `proc format` statement.

```
proc sort data=begin; by sp; run; /* sort dataset by sp                      */
data _null_;                      /* no data set will be created            */
  set begin end=end;              /* variable end to identify end of dataset  */
  by sp;                          /* work by values of the sp variable      */
  file 'sp.lst';                  /* create a file called sp.lst            */
  if last.sp then do;             /* if last obs in sp group then do        */
   spn+1;                         /* number the species                     */
   if spn = 1 then put #1 'proc format;'   /* If first species then begin format stmt*/
   #2 @3 'value splst' @17 spn ' = ' "'"  sp  "'" ;   /* and create label for species*/
 else if end then  put @17 spn ' = ' "'"  sp  "' ;" /'run;' ; /* If last then add end*/
 else              put @17 spn ' = ' "'"  sp  "'" ;   /* Make label for other species*/
end;  run;
%include 'sp.lst';                 * now include and run the proc format statement   */
proc sort data=begin; by plot sp; /* sort dataset by plot and sp            */
data allcat;                      /* create a new dataset called allcat     */
  do plot = 1 to 3;               /* create plot values                     */
     do spnum = 1 to 5;           /* create spnum value (must know how many)  */
        sp = put(spnum,splst.);               /* create sp variable         */
        drop spnum;
     output;  end; end;           /* output obs and end both do loops       */
run;
data both;                        /* create a new dataset called both       */
  merge allcat begin;             /* merge previous datasets allcat and begin */
  by plot sp;                     /* match obs by plot and sp values        */
  if cov = . then cov = 0;        /* if cov is missing set to zero          */
run;  proc print; run;            /* run and print the data                 */
```

The output is the same as before but the file `sp.lst` now contains:

```
proc format;
  value splst   1  = 'A '
                2  = 'B '
                3  = 'C '
                4  = 'D '
                5  = 'E ' ;
run;
```

Contact: Wendy Bergerud
387-5676

NEW PROBLEM

How would you change the do statement if the plot values were 1, 3, 4, 5, 6, and 8?