

## Data Set Preparation

Mplus reads only ASCII (text) data. This is inconvenient, but I'm going to give you some good tips and examples on how to prepare data sets for Mplus. There are several points that are helpful to keep in mind to avoid problems reading data out of your statistical package and into Mplus.

1. Remember the exact location of the data file, including the full path specification. I always keep folder names to a minimum, so that minor errors typing complex file paths don't trip me up on the FILE statement in Mplus.
2. Make sure that the list of the variables that you read out of your statistical packages matches exactly the list of variables you give Mplus in the DATA section. Always double check the order the variables were saved in with the order specified in Mplus.
3. If using free format for input, you need numerical values (asterisk or period are also acceptable) to designate missing data for all variables. This means that system missing values must be given a discrete missing value code.
4. It is good practice to open your data set to look at it to make sure it looks ok. You can open the file in any text editor, but I often just open it in Mplus.

## SPSS

In SPSS, there are two ways to create the raw data files—through the menus and with syntax. My preferred method is with syntax, because a) I like to keep a record of the data files I created, 2) I can double check the variable list to verify the variables in my model are the ones I intended to use, and 3) I can copy the variable names into the Mplus program file to save time and avoid typos, incorrectly ordered variable lists, or omitted variables.

### SPSS Using Menus.

1. file -> save as (specify location and filename and uncheck the "write variable names to spreadsheet" checkbox). Make sure that under "Save as type," you choose "Tab-delimited (\*.dat)".
2. click the "variables" button and check the boxes next to the variables you wish to save out. Note that it is often convenient to first click the "drop all" button and then check the subset of variables that you desire, especially when working with a large data set.
3. Click "continue".
4. Click "save".

**SPSS Using Syntax.** The following syntax lines can be used to save the data as tab-delimited text. I use just a simple example with four variables here. The DESCRIPTIVES command issued afterwards is helpful for double checking that the N is the same as that used in Mplus, but it is optional. The MISSING=LISTWISE command is used to check the N in Mplus when listwise deletion is used (discussed later). If a

DESCRIPTIVES or other command is not used, an EXECUTE statement is needed following the SAVE command.

```
RECODE program TO b3p_conf (SYSMIS=-99).  
  
MISSING VALUES program TO b3p_conf (-99,-6 thru -1).  
  
SAVE TRANSLATE OUTFILE='c:\jason\mplus\consult\ehs\temp.dat'  
  /TYPE=TAB /MAP /REPLACE  
  /KEEP=blp_cesd blv3pdet blv3pint blv3pneg .  
  
DESCRIPTIVES VARS=blp_cesd blv3pdet blv3pint blv3pneg  
  /MISSING=LISTWISE.
```

**SAS Using Syntax.** Use of the PUT statement on the DATA step in SAS will generate an ASCII file. The FILE statement is used to designate a location on the hard drive for the new file. The format statement at the end, (F10.6,'09'X), tells SAS to use a column width of 10 with 6 decimal places for all of the variables and to separate the variables with tabs. The latter is needed to avoid rounding. (I did not include any syntax here, but you may need to declare or recode missing values. The default period is acceptable in Mplus, but you must declare it as a missing value).

```
DATA one; SET data.ehs1;  
  
DATA _NULL_; SET work.one;  
  FILE 'c:\jason\mplus\consult\ehs\ehs1sas.dat';  
  PUT (blp_cesd blv3pdet blv3pint blv3pneg) (F10.6,'09'X);  
RUN;
```

Another option in SAS is to use the PROC EXPORT command, but it automatically lists variable names in the first line of the data file (which will cause problems in Mplus), so the file needs to be opened and edited to remove the names. This method also requires that a FORMAT statement be used to prevent rounding.

```
DATA one (keep=blp_cesd blv3pdet blv3pint blv3pneg); SET  
data.ehs1;  
  
IF MISSING(blp_cesd) THEN blp_cesd=-99;  
IF MISSING(blv3pdet) THEN blv3pdet=-99;  
IF MISSING(blv3pint) THEN blv3pint=-99;  
IF MISSING(blv3pneg) THEN blv3pneg=-99;  
  
FORMAT blp_cesd blv3pdet blv3pint blv3pneg 10.6;  
  
PROC EXPORT DATA=one  
OUTFILE='c:\jason\mplus\consult\ehs\temp2.dat' DBMS=DLM REPLACE ;  
run;
```

### Example 1: Reading Data into Mplus

In Example 1 below, I read the data I created with the above examples into Mplus.

If you have user-defined missing values, you can identify those in Mplus with the MISSING statement in the VARIABLE section. The following are acceptable: MISSING = \*; MISSING = .; MISSING = BLANK; MISSING = varname(#); In the last example, “varname” is any variable name and # is the value in the data set that indicates missing where you can specify multiple discrete values or a range of values.

If using free format for input, as I illustrate below, you cannot use blanks to represent missing data in SPSS. An asterisk, period, or numerical value must be used.

The TYPE=BASIC command is not required, but generates some descriptive data useful for verifying that you have read the data correctly.

```
TITLE:  Example 1, Reading in raw data;

DATA:  FILE=ex1.dat;
       FORMAT=FREE;

VARIABLE:  NAMES = b1p_cesd b1v3pdet b1v3pint b1v3pneg;
           MISSING = b1p_cesd-b1v3pneg(-99,-6--1);

! The following TYPE=BASIC command gives descriptive data
! and is a good idea for checking to make sure the data
! are read in correctly.

ANALYSIS:  TYPE=BASIC;
```

The path on the FILE statement can be abbreviated (e.g., FILE=ex1.dat) as long as the data file resides in the same folder as the input file. Otherwise, the full path is required (e.g., c:\jason\mplus\ehs\ex2.dat).