The SPSS uses an MCMC algorithm known as fully conditional specification (FCS) or chained equations imputation.

The basic idea is to impute incomplete variables one at a time, using the filled-in variable from one step as a predictor in all subsequent steps.

SPSS uses linear regression for continuous variables and logistic regression for categorical variables.
EATING DISORDER RISK DATA

- Questionnaire data from a study of eating disorder risk in a sample of 500 college-aged women

- Variables:

  - Body mass index (BMI), 7 questionnaire items measuring body dissatisfaction, 6 questionnaire items measuring eating disorder risk, binary indicator of past sexual abuse history (0 = no abuse history, 1 = abuse history)

  - All questionnaire items measured on a 7-point Likert scale

ANALYSIS MODEL

- Multiple regression model that predicts eating disorder risk (sum of 6 items) from abuse history and body dissatisfaction (sum of 7 items)

\[
edr = B_0 + B_1(abuse) + B_2(bdscale) + \epsilon\]
DEFINING VARIABLES

• Incomplete variables must be defined as nominal or scale (i.e., continuous) prior to imputation

• SPSS applies linear imputation to scale variables and logistic (or multinomial logistic) regression to categorical variables

• Define variables in the Variable View tab or with syntax

CATEGORICAL VARIABLES

• A multinomail logistic regression model for a Likert outcome has many parameters

• Imputation can be exceedingly slow (the ordinal imputation model in Mplus is much faster)

• Treating ordinal scales as continuous is often fine, but rounding imputed values to the nearest integer can introduce bias
Prior to imputing the data, run an exploratory analysis to assess MCMC convergence.

SPSS provides limited diagnostic information.

To implement my diagnostic macro program (described later), specify imputations = 2 and iterations = 1000.
MULTIPLE IMPUTATION COMMAND

**VARIABLES TAB**

- Select variables, specify the number of imputed data sets, and specify a file name for the imputed data
- To use the diagnostic macro, specify imputations = 2
METHOD TAB

- Select the algorithm (fully conditional specification), and specify the number of between-imputation iterations
- To use the diagnostic macro, specify iterations = 1000

OUTPUT TAB

- Checking the Create iteration history button saves P-step means and standard deviations
- To use the diagnostic macro, save the parameters to a file named parameters.sav
SPSS SYNTAX

/* OPEN DATA */.
get file = 'c:\spss ex\eating risk data.sav'.
dataset name rawdata window = front.

/* DEFINE MEASUREMENT SCALE */.
variable level bmi bds1 to bds7 edr1 to edr6 (scale)
/abuse (nominal).

/* PERFORM EXPLORATORY ANALYSIS */.
dataset activate rawdata.

multiple imputation bmi bds1 to bds7 edr1 to edr6 abuse
/impute method = fcs maxiter = 1000 nimputations = 2
/outfile imputations = temp fcsiterations = 'c:\spss ex\parameters.sav'.

DIAGNOSTIC INFORMATION
(OR LACK THEREOF)

- SPSS provides limited diagnostic information
- The program saves P-step means and standard deviations to a file but does not produce diagnostic information
- I wrote an SPSS macro that creates trace plots and computes the potential scale reduction factor
DIAGNOSTIC MACRO

- `!let !folder` specifies the folder that contains parameters.sav
- `!let !vars` specifies the variables in parameters.sav

```
define diagnosticmacro ()
  */ !FOLDER = FILE PATH TO THE FOLDER CONTAINING THE PARAMETERS.SAV FILE */.
  */ !VARS = VARIABLES IN THE PARAMETERS.SAV FILE */.

!let !folder = 'c:\spss ex'
!let !vars = 'bmi bds1 bds2 bds4 bds7 edr2 edr3 edr5'
```

TRACE PLOT OF THE BMI MEAN

[TRACE PLOT OF MEANS FOR VARIABLE BMI]
• The macro computes the PSR for the P-step means and standard deviations after every 100 iterations

• These parameters often converge quickly than covariances (which SPSS does not provide)

• Be conservative when choosing the between-imputation interval based on the PSR

• The macro displays the maximum PSR for all parameters in a table and in a line graph
After establishing convergence, run MCMC a second time to generate the imputed data sets.

- Imputation details for this example:
  - 20 imputed data sets
  - 200 iterations separating each imputed data set
VARIABLES TAB

METHOD TAB
SPSS stacks the imputed data sets into a single file

A variable named IMPUTATION_ differentiates the data sets

The stacked file format is convenient because data manipulation tasks (e.g., computing new variables, recoding, etc.) need only be executed once

The IMPUTATION_ variable plays an important role in the subsequent analyses …
IMPUTED DATA

COMPUTING SCALE SCORES

- Use the **COMPUTE** command once for each scale score

- Transformations are automatically applied to every data set because the imputed files are stacked
SPLIT FILE COMMAND

- Splitting the file by the IMPUTATION variable invokes the analysis and pooling procedures, if available.

ANALYZING MULTIPLY IMPUTED DATA

- Analyze the data as usual.
- SPSS pools estimates for many common analyses, but not all.
- The program is idiosyncratic in its application of the pooling formulas (e.g., in a regression analysis, SPSS pools the unstandardized coefficients but not the beta weights).
- The icon denotes a procedure that can accommodate multiply imputed data.
/* PERFORM MULTIPLE IMPUTATION ANALYSIS */.

data set activate rawdata.

multiple imputation bmi bds1 to bds7 edr1 to edr6 abuse
/impute method = fcs maxiter = 200 nimputations = 20
/outfile imputations = imputed.

/* COMPUTE SCALE SCORES WITHIN EACH DATA SET */

data set activate imputed.

compute bodydis = sum(bds1 to bds7).
compute eatrisk = sum(edr1 to edr6).
exe.

/* SPLIT THE FILE BY VARIABLE IMPUTATION_ */

sort cases by imputation_.
split file layered by imputation_.

SPSS SYNTAX
SPSS OUTPUT

• SPSS reports the analysis results separately for each imputed data set even when it does not pool the estimates

• The pooled estimates and standard errors appear at the bottom of each table, when available

• Some estimates get pooled, some do not …
CORRELATIONS

<table>
<thead>
<tr>
<th>Imputation Number</th>
<th>Pearson Correlation</th>
<th>eatmk</th>
<th>bodids</th>
<th>abuse</th>
<th>sig (1-tailed)</th>
<th>eatmk</th>
<th>bodids</th>
<th>abuse</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.109</td>
<td>.548</td>
<td>1.000</td>
<td>.309</td>
<td>.348 1.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.500</td>
</tr>
<tr>
<td>19</td>
<td>.644</td>
<td></td>
<td>1.000</td>
<td>.344</td>
<td>.344 1.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.500</td>
</tr>
<tr>
<td>20</td>
<td>.642</td>
<td></td>
<td>.345</td>
<td>1.000</td>
<td>.345 1.000</td>
<td>.000</td>
<td>.000</td>
<td>.000</td>
<td>.500</td>
</tr>
</tbody>
</table>

REGRESSION

<table>
<thead>
<tr>
<th>Imputation Number</th>
<th>Model</th>
<th>Unstandardized Coefficients</th>
<th>Standardized Coefficients</th>
<th>Std. Error</th>
<th>Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>1.217</td>
<td>.474</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>1.564</td>
<td>.572</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>1.871</td>
<td>.511</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.011</td>
<td>.511</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>2.170</td>
<td>.563</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.306</td>
<td>.590</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>2.459</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.504</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>2.549</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.615</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>2.671</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.734</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>2.870</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>2.933</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>3.000</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>3.066</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>3.131</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>3.201</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>3.271</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>3.347</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>bodids</td>
<td></td>
<td>3.425</td>
<td>.596</td>
</tr>
<tr>
<td>1</td>
<td>c(Cont)</td>
<td>abuse</td>
<td></td>
<td>3.507</td>
<td>.596</td>
</tr>
</tbody>
</table>