DEALING WITH MISSING DATA IN EDUCATIONAL RESEARCH

SOFTWARE TUTORIALS

CRAIG K. ENDERS
REMUS MITCHELL
MICHAEL P. WOLLER

DEALING WITH MISSING DATA IN EDUCATIONAL RESEARCH

Software Tutorials

Craig K. Enders
Remus Mitchell
Michael P. Woller

The manuscript was developed as part of a missing data toolkit supported by Institute of Educational Sciences award R305D22000

Contents

Contents i

Data File Descriptions 1

Getting Started With Software 6

FIML: Regression With Multivariate Normal Data 7

1.1 Analysis Model 7
1.2 Mplus FIML Script 8
1.3 Mplus Output 9
1.4 R lavaan FIML Script 13
1.5 R Output 14
1.6 Adding Auxiliary Variables 17
1.7 Mplus FIML Script and Output 18
1.8 R lavaan FIML Script and Output 20

FIML: Binary Logistic Regression 21

2.1 Analysis Model 21
2.2 Mplus FIML Script 22
2.3 Mplus Output 23
2.4 Adding Auxiliary Variables 26
2.5 Mplus FIML Script 27
2.6 R mdmb FIML Script 27
2.7 R Output 30

FIML: Regression With Binary and Ordinal Predictors **32**

3.1 Analysis Model 32
3.2 Mplus FIML Script 34
3.3 Mplus Output 35
3.4 R mdmb FIML Script 37
3.5 R Output 40

FIML: Moderated Regression With an Interaction **42**

4.1 Analysis Model 42
4.2 R mdmb FIML Script 44
4.3 R Output 47

FIML: Curvilinear Regression **48**

5.1 Analysis Model 48
5.2 R mdmb FIML Script 50
5.3 R Output 53

MCMC: Regression With Multivariate Normal Data **54**

6.1 Analysis Model 54
6.2 Blimp and rblimp MCMC Scripts 55
6.3 Blimp and rblimp Output 58
6.4 Saving Model-Based Multiple Imputations 63
6.5 Analyzing Multiple Imputations in R 65
6.6 R Output 66
6.7 Analyzing Multiple Imputations in Mplus 67
6.8 Mplus Output 69
6.9 Analyzing Multiple Imputations in SPSS 71
6.10 SPSS Output 72

MCMC: Binary Logistic Regression **74**

7.1 Analysis Model 74
7.2 Blimp and rblimp MCMC Scripts 75
7.3 Blimp and rblimp Output 77
7.4 Saving Model-Based Multiple Imputations 81

7.5 Analyzing Multiple Imputations in R 83
7.6 R Output 84
7.7 Analyzing Multiple Imputations in Mplus 85
7.8 Mplus Output 86
7.9 Analyzing Multiple Imputations in SPSS 88
7.10 SPSS Output 89

MCMC: Regression With Binary and Ordinal Predictors 91

8.1 Analysis Model 91
8.2 Blimp and rblimp MCMC Scripts 92
8.3 Blimp and rblimp Output 95
8.4 Saving Model-Based Multiple Imputations 98
8.5 Analyzing Multiple Imputations in R 100
8.6 R Output 101
8.7 Analyzing Multiple Imputations in Mplus 103
8.8 Mplus Output 104
8.9 Analyzing Multiple Imputations in SPSS 105
8.8 SPSS Output 106

MCMC: Regression With Multicategorical Predictors 108

9.1 Analysis Model 109
9.2 Blimp and rblimp MCMC Scripts 109
9.3 Blimp and rblimp Output 112
9.4 Saving Model-Based Multiple Imputations 114
9.5 Analyzing Multiple Imputations in R 117
9.6 R Output 118
9.7 Analyzing Multiple Imputations in Mplus 119
9.8 Mplus Output 120
9.9 Analyzing Multiple Imputations in SPSS 122
9.10 SPSS Output 123

MCMC: Moderated Regression With an Interaction 125

10.1 Analysis Model 125
10.2 Blimp and rblimp MCMC Scripts 126
10.3 Blimp and rblimp Output 129

10.4 Saving Model-Based Multiple Imputations 132
10.5 Analyzing Multiple Imputations in R 134
10.6 R Output 136
10.7 Analyzing Multiple Imputations in Mplus 138
10.8 Mplus Output 140
10.9 Analyzing Multiple Imputations in SPSS 141
10.10 SPSS Output 143

MCMC: Curvilinear Regression 145

11.1 Analysis Model 145
11.2 Blimp and rblimp MCMC Scripts 146
11.3 Blimp and rblimp Output 149
11.4 Saving Model-Based Multiple Imputations 151
11.5 Analyzing Multiple Imputations in R 153
11.6 R Output 154
11.7 Analyzing Multiple Imputations in Mplus 155
11.8 Mplus Output 157
11.9 Analyzing Multiple Imputations in SPSS 158
11.10 SPSS Output 159

FCS Multiple Imputation: Paired-Samples Comparison 161

12.1 Imputation and Analysis Models 161
12.2 Blimp and rblimp FCS Scripts 162
12.3 Blimp and rblimp Output 164
12.4 Analyzing Multiple Imputations in R 167
12.5 R Output 168
12.6 Analyzing Multiple Imputations in Mplus 168
12.7 Mplus Output 170
12.8 Analyzing Multiple Imputations in SPSS 170
12.9 SPSS Output 171

FCS Multiple Imputation: Multiple Regression 173

13.1 Imputation and Analysis Models 173
13.2 Blimp and rblimp FCS Scripts 174
13.3 Blimp and rblimp Output 176

13.4 Analyzing Multiple Imputations in R 179
13.5 R Output 180
13.6 Analyzing Multiple Imputations in Mplus 181
13.7 Mplus Output 182
13.8 Analyzing Multiple Imputations in SPSS 184
13.9 SPSS Output 185

FCS Multiple Imputation: Regression with a Multicategorical Predictor 187

14.1 Imputation and Analysis Models 188
14.2 Blimp and rblimp FCS Scripts 188
14.3 Blimp and rblimp Output 191
14.4 Analyzing Multiple Imputations in R 193
14.5 R Output 195
14.6 Analyzing Multiple Imputations in Mplus 196
14.7 Mplus Output 197
14.8 Analyzing Multiple Imputations in SPSS 199
14.9 SPSS Output 200

FCS Multiple Imputation: Random Intercept Model 202

15.1 Imputation and Analysis Models 202
15.2 Blimp and rblimp FCS Scripts 203
15.3 Blimp and rblimp Output 206
15.4 Analyzing Multiple Imputations in R 208
15.5 R Output 209
15.6 Analyzing Multiple Imputations in Mplus 211
15.7 Mplus Output 212
15.8 Analyzing Multiple Imputations in SPSS 213
15.9 SPSS Output 215

MCMC: Random Intercept Model 217

16.1 Analysis Model 217
16.2 Blimp and rblimp MCMC Scripts 218
16.3 Blimp and rblimp Output 220
16.4 Saving Model-Based Multiple Imputations 223
16.5 Analyzing Multiple Imputations in R 226

16.6 R Output 227
16.7 Analyzing Multiple Imputations in Mplus 228
16.8 Mplus Output 229
16.8 Analyzing Multiple Imputations in SPSS 231
16.9 SPSS Output 232

MCMC: Random Slope Model With an Interaction 234

17.1 Analysis Model 234
17.2 Blimp and rblimp MCMC Scripts 235
17.3 Blimp and rblimp Output 237
17.4 Saving Model-Based Multiple Imputations 241
17.5 Analyzing Multiple Imputations in R 244
17.6 R Output 245
17.7 Analyzing Multiple Imputations in Mplus 248
17.8 Mplus Output 249
17.9 Analyzing Multiple Imputations in SPSS 251
17.10 SPSS Output 252

MCMC: Three-Level Growth Model 255

18.1 Analysis Model 256
18.2 Blimp and rblimp MCMC Scripts 256
18.3 Blimp and rblimp Output 258
18.4 Saving Model-Based Multiple Imputations 262
18.5 Analyzing Multiple Imputations in R 265
18.6 R Output 266
18.7 Analyzing Multiple Imputations in Mplus 268
18.8 Mplus Output 270
18.9 Analyzing Multiple Imputations in SPSS 272
18.10 SPSS Output 273

FIML and MCMC: Selection Model for Regression 276

19.1 Analysis Model 277
19.2 Mplus FIML Script 278
19.3 Mplus Output 280
19.4 Blimp and rblimp MCMC Scripts 282

19.5 Blimp and rblimp Output 284

FIML and MCMC: Pattern Mixture Model For Regression 289

20.1 Analysis Model 290
20.2 Mplus FIML Script 292
20.3 Mplus Output 294
20.4 Blimp and rblimp MCMC Scripts 297
20.5 Blimp and rblimp Output 300

References 304

Data File Descriptions

The analysis examples use synthetic data sets created to closely resemble those from educational studies described in Montague et al. (2005) and Montague et al. (2014). The data and analysis scripts are available for download from the project website: www.appliedmissingdata.com/videos.

The `behaviorachievement.dat` file is taken from a longitudinal study that followed 138 students from primary to middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade.

The `mathachievement.dat` data set is taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables.

The `problemsolving2level.dat` data set is taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 982 student-level records include pretest and posttest math problem-solving and self-efficacy scores, standardized math scores taken from a statewide assessment, and several sociodemographic variables.

The `problemsolving3level.dat` data set is taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 6874 within-subjects data records include seven monthly measures of math problem-solving and self-efficacy, standardized math scores taken from a statewide assessment, and several sociodemographic variables.

Variable Definitions for `behaviorachievement.dat` File

Name	Definition	Missing %	Scale
ID	Individual identifier	0	Integer index
MALE	Gender dummy code	1.5	0 = Female, 1 = Male
HISPANIC	Hispanic dummy code	5.1	0 = African American, 1 = Hispanic
RISKGRP	Emotion/behavior disorder risk	2.2	1 = Low, 2 = Medium, 3 = High
ATRISK	Emotion/behavior disorder risk	2.2	0 = Low, 1 = Medium/high
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric (17 to 92)
$LRNPROB_1$	1st grade learning problems	2.2	Numeric (31 to 88)
$READ_1$	1st grade reading composite	6.5	Numeric (39 to 153)
$READ_2$	2nd grade reading composite	9.4	Numeric (20 to 150)
$READ_3$	3rd grade reading composite	14.5	Numeric (46 to 138)
$READ_9$	9th grade reading composite	17.4	Numeric (41 to 123)
$READGRP_9$	9th grade reading classification	17.4	0 = Below average, 1 = Average
$STANREAD_7$	7th grade standardized reading	19.6	Numeric (100 to 399)
$MATH_1$	1st grade math composite	6.5	Numeric (60 to 149)
$MATH_2$	2nd grade math composite	9.4	Numeric (76 to 138)
$MATH_3$	3rd grade math composite	14.5	Numeric (71 to 143)
$MATH_9$	9th grade math composite	17.4	Numeric (55 to 127)
$MATHGRP_9$	9th grade math classification	17.4	0 = Below average, 1 = Average
$STANMATH_7$	7th grade standardized math	19.6	Numeric (100 to 421)

Variable Definitions for `mathachievement.dat` File

Name	Definition	Missing %	Scale
ID	Individual identifier	0	Integer index
CONDITION	Experimental condition	0	0 = Comparison, 1 = Intervention
MALE	Gender dummy code	0	0 = Female, 1 = Male
FRLUNCH	Lunch assistance dummy code	4.4	0 = None, 1 = Lunch assistance
ATRISK	Emotion/behavior disorder risk	5.2	0 = Low risk, 1 = At-risk
STANREAD	Standardized reading	9.2	Numeric (27 to 69)
EFFICACY	Math self-efficacy rating scale	9.6	Ordinal (1 to 6)
ANXIETY	Math anxiety composite	8.4	Numeric (0 to 44)
MATHPRE	Math achievement pretest	0	Numeric (26 to 76)
MATHPOST	Math achievement posttest	18.0	Numeric (37 to 85)

Variable Definitions for `problemsolving2level.dat` File

Name	Definition	Missing %	Scale
SCHOOL	School identifier	0	Integer index
STUDENT	Student identifier	0	Integer index
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental
TEACHEXP	Teacher years of experience	10.8	Numeric (4.3 to 24.6)
ESLPCT	% English as second language	0	Numeric (10 to 100)
HISPANIC	Ethnicity/race	9.0	0 = White/Black, 1 = Hispanic
MALE	Gender dummy code	0	0 = Female, 1 = Male
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Lunch assistance
LOWACH	Low achievement code	5.2	0 = Typically achieving, 1 = Low achieving
STANMATH	Standardized math scores	7.4	Numeric (5.3 to 87.8)
EFFICACYPRE	Math self-efficacy pretest	0	Numeric (0 to 12)
EFFICACYPST	Math self-efficacy posttest	20.5	Numeric (0 to 12)
PSOLVEPRE	Math problem-solving pretest	0	Numeric (37 to 66)
PSOLVEPST	Math problem-solving posttest	20.5	Numeric (37 to 65)

Variable Definitions for `problemsolving3level.dat` File

Name	Definition	Missing %	Scale
SCHOOL	School identifier	0	Integer index
STUDENT	Student identifier	0	Integer index
WAVE	Monthly wave identifier	0	Integer index (1 to 7)
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental
TEACHEXP	Teacher years of experience	10.8	Numeric (4.3 to 24.6)
ESLPCT	% English as second language	0	Numeric (10 to 100)
HISPANIC	Ethnicity/race	9.0	0 = Non-Hispanic, 1 = Hispanic
MALE	Gender dummy code	0	0 = Female, 1 = Male
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Lunch assistance
LOWACH	Low achievement code	5.2	0 = Typically achieving, 1 = Low achieving
STANMATH	Standardized reading	7.4	Numeric (5.3 to 87.8)
MONTH	Time scores (baseline = 0)	0	Numeric (0 to 6)
$MONTH_7$	Time scores (endpoint = 0)	0	Numeric (–6 to 0)
EFFICACY	Math self-efficacy	11.4	Numeric (0 to 14)
PROBSOLVE	Math problem-solving	11.4	Numeric (37 to 68)

Getting Started With Software

The tutorial examples use the Blimp (Keller & Enders, 2021) application for MCMC estimation and multiple imputation. Blimp's development was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305D150056 & R305D190002 to UCLA. Blimp is freely available at www.appliedmissingdata.com/blimp. The Blimp User Guide is available from the same website and from the `Help > Help` pull-down. Blimp scripts can be executed from the Blimp Studio graphical interface (macOS and Windows), and the `rblimp` package is also available for the R environment (Keller, 2024). The `rblimp` package is currently available for download at Brian Keller's github, and a forthcoming version will be available through CRAN. The standalone version of Blimp must be installed prior to downloading and installing `rblimp`.

The tutorial examples use Mplus for maximum likelihood estimation and for analyzing multiply imputed data sets. A free demo version of Mplus is available at www.statmodel.com/demo.shtml. Many of the scripts run on the demo version, which is limited to six variables. The tutorial examples also use various R packages for maximum likelihood estimation and for analyzing multiply imputed data sets. The installation commands for the R packages used in this document are as follows.

```
install.packages('lavaan', dependencies = T)
install.packages('semTools', dependencies = T)
install.packages('rockchalk', dependencies = T)
install.packages('mitml', dependencies = T)
install.packages('mdmb', dependencies = T)
install.packages('remotes', dependencies = T)
remotes::install_github('bkeller2/fdir')
remotes::install_github('blimp-stats/rblimp')
```

1

FIML: Regression With Multivariate Normal Data

This example illustrates a multiple regression analysis with multivariate normal incomplete data. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric

1.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) + \varepsilon \quad (1)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. The Mplus and R scripts below assign a multivariate normal distribution to the set of analysis variables.

1.2 Mplus FIML Script

The code block below shows Mplus script `Ex1.1.inp`.

Mplus Script Ex1.1.inp

```
1   DATA:
2   file = behaviorachievement.dat;
3   VARIABLE:
4   names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
5     read1 read2 read3 read9 read9grp stanread7
6     math1 math2 math3 math9 math9grp stanmath7;
7   usevariables = read9 read1 lrnprob1 behsymp1;
8   missing = all(999);
9   ANALYSIS:
10  estimator = ml;
11  MODEL:
12  read1 lrnprob1 behsymp1;
13  read9 on read1 lrnprob1 behsymp1 (beta1-beta3);
14  MODEL TEST:
15  0 = beta1; 0 = beta2; 0 = beta3;
16  OUTPUT:
17  patterns sampstat stdyx cinterval;
```

The `DATA` command specifies the name of the input text file. No file path is required when the data set is in the same directory as the script, as it is here. The `VARIABLE` command provides information about the data. Beginning on line 4, the `names` subcommand assigns names to the variables in the input data, the `usevariables` subcommand selects variables for the analysis, and the `missing` subcommand gives the global missing value code. The `ANALYSIS` command and `estimator` subcommand specify full information maximum likelihood estimation. These commands are optional because the maximum likelihood missing data handling is the default. If the variables are nonnormal, specifying `estimator = mlr` on line 10 generates robust test statistics and standard errors.

The MODEL section of the script consists of two lines. Listing all predictors by name on line 12 is important because doing so invokes a multivariate normal distribution for these variables. As mentioned previously, assigning distributional assumptions to predictors is necessary for missing data handling. On line 13, the outcome variable appears to the left of the on keyword, and the predictors appear to the right. The end of this line includes labels for the slope parameters in parentheses. The subsequent MODEL TEST command uses these labels to specify a custom significance test of the omnibus null hypothesis that all three population slopes equal zero. Finally, the OUTPUT command specifies four keywords on line 17 that request a summary of the missing data patterns, maximum likelihood estimates of sample statistics, standardized coefficients, and confidence intervals.

1.3 Mplus Output

Information about the missing data patterns is found near the top of the output file. The table in the excerpt below shows the analysis variables in the rows and missing data patterns in the columns. The output also displays the frequency of each missing data pattern.

```
SUMMARY OF MISSING DATA PATTERNS


     MISSING DATA PATTERNS (x = not missing)

               1  2  3  4  5  6  7
 READ9         x  x  x  x
 READ1         x  x  x     x  x
 LRNPROB1      x  x     x  x  x  x
 BEHSYMP1      x     x  x  x     x


     MISSING DATA PATTERN FREQUENCIES

    Pattern   Frequency     Pattern   Frequency     Pattern   Frequency
          1          99           4           8           7           1
          2           4           5          22
          3           3           6           1
```

Next, the covariance coverage matrix displays the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. A low value on the off-diagonal indicates that the data contain little information about a bivariate association.

```
COVARIANCE COVERAGE OF DATA

Minimum covariance coverage value   0.100

     PROPORTION OF DATA PRESENT

           Covariance Coverage
              READ9         READ1         LRNPROB1      BEHSYMP1
              ________      ________      ________      ________
 READ9          0.826
 READ1          0.768         0.935
 LRNPROB1       0.804         0.913         0.978
 BEHSYMP1       0.797         0.899         0.942         0.964
```

The `MODEL TEST` command in the previous script requested an analogous Wald chi-square statistic that evaluates the null hypothesis that all population slopes equal zero. The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       14
...

Wald Test of Parameter Constraints

          Value                            159.666
          Degrees of Freedom                     3
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. Because the analysis specifies a multivariate normal distribution for the predictors, the means, variances, and covariances of these variables are printed along with the focal model estimates. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown in bold typeface.

MODEL RESULTS

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value
READ9 ON				
READ1	**0.503**	**0.045**	**11.230**	**0.000**
LRNPROB1	**-0.224**	**0.132**	**-1.703**	**0.089**
BEHSYMP1	**-0.222**	**0.110**	**-2.023**	**0.043**
LRNPROB1 WITH				
READ1	-5.643	19.063	-0.296	0.767
BEHSYMP1 WITH				
READ1	-11.235	20.841	-0.539	0.590
LRNPROB1	92.048	13.548	6.794	0.000
Means				
READ1	86.732	1.709	50.739	0.000
LRNPROB1	52.328	0.914	57.224	0.000
BEHSYMP1	49.483	1.039	47.631	0.000
Intercepts				
READ9	**66.901**	**6.465**	**10.349**	**0.000**

```
Variances
    READ1            387.270     48.040      8.061      0.000
    LRNPROB1         114.162     13.820      8.260      0.000
    BEHSYMP1         146.318     17.738      8.249      0.000

 Residual Variances
    READ9             86.095     11.813      7.288      0.000
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.50 points on the outcome. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 11.23$, $p < .001$).

Specifying the stdyx keyword as an option prints the table of standardized estimates and *R*-squared statistic shown below. The slope coefficients convey the expected change in standard deviation units for a one standard deviation increase in each predictor. For example, the model predicts that two individuals who differ by one standard deviation on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.68 standard deviations on the outcome. The *R*-squared statistic at the bottom of this section indicates that the collection predictors explain 59% of the variation in ninth-grade reading scores.

```
STANDARDIZED MODEL RESULTS

STDYX Standardization

                                                    Two-Tailed
                  Estimate       S.E.  Est./S.E.    P-Value

 READ9    ON
    READ1            0.683      0.049     13.901      0.000
    LRNPROB1        -0.165      0.097     -1.698      0.089
    BEHSYMP1        -0.185      0.091     -2.032      0.042

 LRNPROB1 WITH
    READ1           -0.027      0.091     -0.296      0.767
```

```
BEHSYMP1 WITH
    READ1             -0.047      0.087     -0.541      0.588
    LRNPROB1           0.712      0.042     16.784      0.000

 Means
    READ1              4.407      0.287     15.339      0.000
    LRNPROB1           4.897      0.309     15.864      0.000
    BEHSYMP1           4.091      0.262     15.594      0.000

 Intercepts
    READ9              4.620      0.575      8.032      0.000

 Variances
    READ1              1.000      0.000    999.000    999.000
    LRNPROB1           1.000      0.000    999.000    999.000
    BEHSYMP1           1.000      0.000    999.000    999.000

 Residual Variances
    READ9              0.411      0.059      6.974      0.000

R-SQUARE

    Observed                                        Two-Tailed
    Variable        Estimate       S.E.  Est./S.E.    P-Value

    READ9              0.589      0.059     10.014      0.000
```

1.4 R lavaan FIML Script

The R input file for the analysis is `Ex1.1.R`. The example requires the `lavaan` package.

R Script Ex1.1.R

```
1   library(lavaan)
2   load('behaviorachievement.rda')
3
4   model <- 'read9 ~ b1*read1 + b2*lrnprob1 + b3*behsymp1'
5   fit <- sem(model, behaviorachievement, fixed.x = F, missing = 'fiml')
```

```
6
7    inspect(fit, 'patterns')
8    inspect(fit, 'coverage')
9    summary(fit, rsquare = T, standardize = T)
10
11   wald.constraints <- 'b1 == 0; b2 == 0; b3 == 0;'
12   lavTestWald(fit, constraints = wald.constraints)
```

The `model` variable on line 4 defines a text string specifying the regression model, with the outcome variable on the left side of the tilde and the predictors to the right. Each predictor's slope is preceded by a label (i.e., `b1`, `b2`, and `b3`). A subsequent command uses these labels to specify a custom significance test of the null hypothesis that the population slopes equal zero. On line 5, the model string and data frame are passed into the `sem` function. The `fixed.x = F` parameter specifies that the predictors are treated as normally distributed variables, and `missing = 'fiml'` requests missing data estimation. The `fixed.x` specification is important because it invokes a multivariate normal distribution for the analysis variables. As mentioned previously, assigning distributions to incomplete predictors is necessary for missing data handling.

The `inspect` functions on lines 7 and 8 produce a table of missing data patterns and a covariance coverage matrix with the proportion of observed data for each variable or variable pair, respectively. The `summary` function on line 9 produces tabular results with standardized estimates and the *R*-squared statistic. Finally, the `wald.constraints` variable on line 11 defines a text string that uses the aforementioned labels to specify the null hypothesis that all three population slopes equal zero. The `lavTestWald` function on line 12 uses that text string to generate a chi-square statistic, degrees of freedom, and *p*-value.

1.5 R Output

The `inspect` functions in the previous script request information about the missing data patterns and missing data rates. The missing data pattern table in output below shows the analysis variables in the columns and missing data patterns in the rows (1 = observed, 0 = missing).

```
      read9 read1 lrnpr1 bhsym1
[1,]     1     1      1      1
[2,]     0     1      1      1
[3,]     1     0      1      1
[4,]     1     1      1      0
```

```
[5,]    1     1      0      1
[6,]    0     0      1      1
[7,]    0     1      1      0
```

The covariance coverage matrix displays the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. A low value on the off-diagonal indicates that the data contain little information about a bivariate association.

```
         read9 read1 lrnpr1 bhsym1
read9    0.826
read1    0.768 0.935
lrnprob1 0.804 0.913 0.978
behsymp1 0.797 0.899 0.942  0.964
```

The table of parameter estimates is shown below. Because the analysis specifies a multivariate normal distribution for the predictors, the means, variances, and covariances of these variables are printed along with the focal model estimates. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The rightmost column gives the standardized coefficients. The focal model results are shown in bold typeface.

```
Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  read9 ~
    read1     (b1)    0.503    0.045   11.230    0.000    0.503    0.683
    lrnprob1  (b2)   -0.224    0.132   -1.702    0.089   -0.224   -0.165
    behsymp1  (b3)   -0.222    0.110   -2.023    0.043   -0.222   -0.185

Covariances:
                   Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
  read1 ~~
    lrnprob1         -5.637   19.063   -0.296    0.767   -5.637   -0.027
    behsymp1        -11.228   20.841   -0.539    0.590  -11.228   -0.047
  lrnprob1 ~~
```

```
    behsymp1         92.048   13.548    6.794    0.000   92.048    0.712

Intercepts:
                  Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
   .read9           66.901    6.465   10.349    0.000   66.901    4.620
    read1           86.732    1.709   50.739    0.000   86.732    4.407
    lrnprob1        52.328    0.914   57.225    0.000   52.328    4.897
    behsymp1        49.483    1.039   47.631    0.000   49.483    4.091

Variances:
                  Estimate  Std.Err  z-value  P(>|z|)   Std.lv  Std.all
   .read9           86.096   11.813    7.288    0.000   86.096    0.411
    read1          387.275   48.041    8.061    0.000  387.275    1.000
    lrnprob1       114.160   13.820    8.260    0.000  114.160    1.000
    behsymp1       146.317   17.738    8.249    0.000  146.317    1.000

R-Square:
                  Estimate
    read9            0.589
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.50 points on the outcome. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 11.23$, $p < .001$).

The standardized coefficients in the Std.all column convey the expected change in standard deviation units for a one standard deviation increase in each predictor. For example, the model predicts that two individuals who differ by one standard deviation on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.68 standard deviations on the outcome. The *R*-squared statistic at the bottom of this section indicates that the collection predictors explain 59% of the variation in ninth-grade reading scores.

Most software programs that fit regression models report an omnibus *F* test that evaluates the set of slope coefficients. The lavTestWald function in the previous script requested an analogous Wald chi-square statistic that evaluates the null hypothesis that all population slopes equal zero. The chi-square statistic, degrees of freedom, and *p*-value appear on the output as follows. The test statistic is statistically significant, thus refuting the null hypothesis.

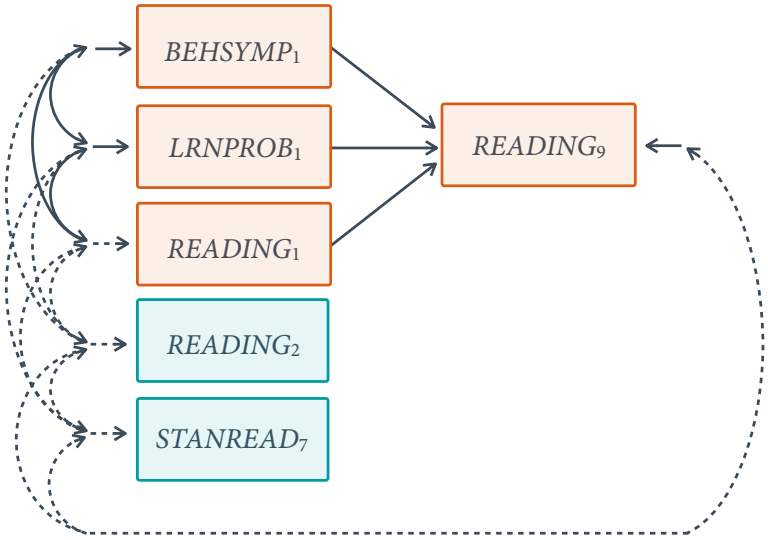
```
$stat
[1] 159.6636

$df
[1] 3

$p.value
[1] 0
```

1.6 Adding Auxiliary Variables

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). The next part of example introduces two auxiliary variables using the saturated correlates approach described by Graham (2003). The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

A path diagram of the saturated correlates model is shown below. The curved arrows depict correlations and residual correlations that connect the auxiliary variables to each other and to the residuals of the focal variables. Both Mplus and R have facilities that automatically introduce auxiliary variables according to this model. Note that the saturated correlates approach assumes that all variables are multivariate normal.



1.7 Mplus FIML Script and Output

The code block below shows Mplus script `Ex1.2.inp`. The only change to the script is the `auxiliary` subcommand on line 8, which functions as a second variable list containing just the auxiliary variables. The `(m)` specification indicates that the additional variables are missing data auxiliary variables (Mplus uses this command for other purposes unrelated to missing data). Finally, note that the additional variables are omitted from the `usevariables` line.

Mplus Script Ex1.2.inp

```
1   DATA:
2   file = behaviorachievement.dat;
3   VARIABLE:
4   names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
5     read1 read2 read3 read9 read9grp stanread7
6     math1 math2 math3 math9 math9grp stanmath7;
7   usevariables = read9 read1 lrnprob1 behsymp1;
8   auxiliary = (m) read2 stanread7;
9   missing = all(999);
10  ANALYSIS:
11  estimator = ml;
12  MODEL:
```

```
13   read1 lrnprob1 behsymp1;
14   read9 on read1 lrnprob1 behsymp1 (beta1-beta3);
15   MODEL TEST:
16   0 = beta1; 0 = beta2; 0 = beta3;
17   OUTPUT:
18   patterns sampstat stdyx cinterval;
```

The only indication that auxiliary variables are included in the model appears in the SUMMARY OF ANALYSIS table near the top of the output file. The main body of the output doesn't change with auxiliary variables, as the additional parameters (e.g., the curved arrows, or correlations) are suppressed. The estimates and standard errors may change, which is expected when including auxiliary variables that have salient semipartial correlations with the incomplete variables.

```
SUMMARY OF ANALYSIS

Number of groups                                         1
Number of observations                                 138

Number of dependent variables                            1
Number of independent variables                          3
Number of continuous latent variables                    0



Observed dependent variables

  Continuous
   READ9

Observed independent variables
   READ1       LRNPROB1    BEHSYMP1

Observed auxiliary variables
   READ2       STANREAD7
```

1.8 R lavaan FIML Script and Output

The R input file that incorporates auxiliary variables is `Ex1.2.R`. The example requires the `lavaan` and semTools packages.

R Script Ex1.2.R

```
library(lavaan)
load('behaviorachievement.rda')

model <- 'read9 ~ b1*read1 + b2*lrnprob1 + b3*behsymp1'
fit <- sem.auxiliary(model, behaviorachievement, fixed.x = F,
  aux = c('read2','stanread7'))

inspect(fit, 'patterns')
inspect(fit, 'coverage')
summary(fit, rsquare = T, standardize = T)
```

The `model` text string remains the same with auxiliary variables. The major change is that the `sem.auxiliary` function replaces the `sem` function in `Ex1.1.R`. The `aux` parameter defines a vector of auxiliary variable names for the saturated correlates model. Unlike Mplus, the R output includes the auxiliary variable parameters. The additional estimates can be ignored because they are not the substantive focus.

2

FIML: Binary Logistic Regression

This example illustrates a binary logistic regression analysis with incomplete data. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READGRP_9$	9th grade reading classification	17.4	0 = Below average, 1 = Average
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

2.1 Analysis Model

The analysis model features a binary classification of ninth grade reading performance regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$\text{logit}(READGRP_9) = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) \quad (2)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Models with mixtures of categorical and numeric variables require a factored regression specification that separates the likelihood function into separate components for each variable type. Mplus assigns a multivariate normal distribution to the predictors, whereas the R script links predictors to one another using a sequence of univariate regression models.

2.2 Mplus FIML Script

The code block below shows Mplus script `Ex2.1.inp`.

Mplus Script Ex2.1.inp

```
DATA:
file = behaviorachievement.dat;
VARIABLE:
names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
  read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7;
usevariables = read9grp read1 lrnprob1 behsymp1;
categorical = read9grp;
missing = all(999);
ANALYSIS:
estimator = ml;
link = logit;
integration = montecarlo;
MODEL:
read1 lrnprob1 behsymp1;
read9grp on read1 lrnprob1 behsymp1 (beta1-beta3);
MODEL TEST:
0 = beta1; 0 = beta2; 0 = beta3;
OUTPUT:
patterns sampstat stdyx cinterval;
```

The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. The VARIABLE command provides information about the data. Beginning on line 4, the names subcommand assigns names to the variables in the input data file, the usevariables subcommand selects variables for the analysis, and the missing subcommand gives the global missing value code. The categorical subcommand on line 8 defines the outcome as a binary variable. The ANALYSIS command and estimator subcommand specify full information maximum likelihood estimation. Additionally, the link = logit subcommand specifies a logistic regression for the outcome variable, and integration = montecarlo invokes an algorithmic method for models with mixed variable types.

The MODEL section of the script consists of two lines. Listing all predictors by name on line 15 is important because doing so invokes a multivariate normal distribution for these variables. As mentioned previously, assigning distributional assumptions to predictors is necessary for missing data handling. On line 16, the outcome variable appears to the left of the on keyword, and the predictors appear to the right. The end of this line includes labels for the slope parameters in parentheses. The subsequent MODEL TEST command uses these labels to specify a custom significance test of the omnibus null hypothesis that all three population slopes equal zero. Finally, the OUTPUT command specifies four keywords on line 20 that request a summary of the missing data patterns, maximum likelihood estimates of sample statistics, standardized coefficients, and confidence intervals.

2.3 Mplus Output

Information about the missing data patterns is found near the top of the output file. Following the missing data pattern table, the output displays a covariance coverage matrix that gives the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. The format of these tables is the same as those shown in Section 1.3.

The MODEL TEST command in the previous script requested an analogous Wald chi-square statistic that evaluates the null hypothesis that all population slopes equal zero. The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the MODEL FIT INFORMATION section under the Wald Test of Parameter Constraints heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       13

...

Wald Test of Parameter Constraints

          Value                            21.889
          Degrees of Freedom                    3
          P-Value                          0.0001
```

The table of unstandardized parameter estimates is shown below. Because the analysis specifies a multivariate normal distribution for the predictors, the means, variances, and covariances of these variables are printed along with the focal model estimates. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown in bold typeface.

MODEL RESULTS

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value
READGRP9 ON				
READ1	**0.069**	**0.016**	**4.446**	**0.000**
LRNPROB1	**-0.018**	**0.033**	**-0.549**	**0.583**
BEHSYMP1	**-0.028**	**0.028**	**-1.014**	**0.311**
LRNPROB1 WITH				
READ1	3.085	19.553	0.158	0.875
BEHSYMP1 WITH				
READ1	-5.194	21.046	-0.247	0.805
LRNPROB1	92.088	13.554	6.794	0.000

```
Means
   READ1              86.974      1.719     50.598      0.000
   LRNPROB1           52.319      0.914     57.267      0.000
   BEHSYMP1           49.488      1.041     47.544      0.000

Thresholds
   READGRP9$1          3.874      1.729      2.240      0.025

Variances
   READ1             384.526     47.859      8.035      0.000
   LRNPROB1          113.906     13.775      8.269      0.000
   BEHSYMP1          146.740     17.818      8.235      0.000
```

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 differ by 0.07. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 4.45$, $p < .001$). Note that Mplus reports a threshold parameter instead of the usual regression intercept. The threshold from a binary logistic model has the same value as the intercept but the opposite sign (i.e., $\hat{\beta}_0 = -3.87$).

Finally, the printed output also includes the table of odds ratios that reflect multiplicative changes to the odds. For example, a one-point increase in first grade reading scores increases the odds of achieving an average ninth grade reading level by a factor 1.07, holding first grade learning problems and behavioral symptoms constant.

```
LOGISTIC REGRESSION ODDS RATIO RESULTS

                                                 95% C.I.
                  Estimate       S.E.  Lower 2.5% Upper 2.5%

READGRP9    ON
   READ1             1.072      0.017      1.040      1.105
   LRNPROB1          0.982      0.032      0.921      1.047
   BEHSYMP1          0.972      0.027      0.921      1.027
```

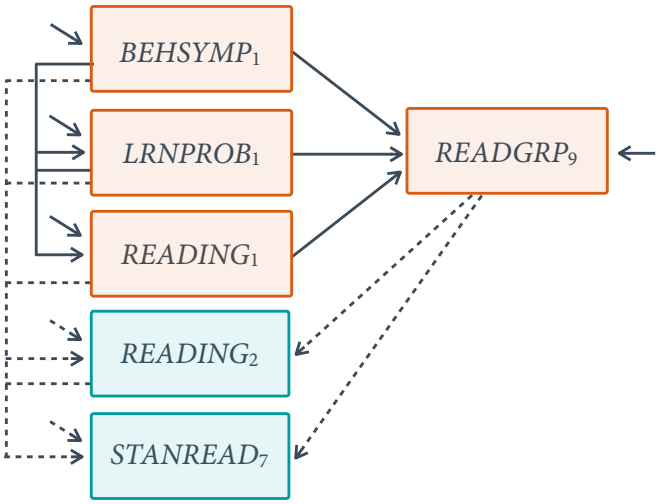
2.4 Adding Auxiliary Variables

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). The saturated correlates model from Section 1.6 is not applicable to logistic regression models because it assumes multivariate normality. Instead, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other.

The additional regression equations are as follows.

$$\begin{aligned} READ_2 &= \gamma_{01} + \gamma_{11}(READGRP_9) + \gamma_{21}(READ_1) + \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1) + \epsilon_1 \\ STANREAD_7 &= \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READGRP_9) + \gamma_{32}(READ_1) \\ &\quad + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1) + \epsilon_2 \end{aligned} \tag{3}$$

Along with the logistic regression model from Equation 2, the collection of regression equations can be viewed as the path model shown below, where the dashed lines are the additional regressions. With this method, the focal model is one part of a larger network of variables. Importantly, the path model does not represent substantive theory, but is simply a tool for linking the auxiliary variables to the focal variables and to each other.



2.5 Mplus FIML Script

The code block below shows an excerpt from Mplus script `Ex2.2.inp`. The `MODEL` command includes two new regression equations, but the script is otherwise similar to `Ex2.1.inp`.

```
MODEL:
read1 lrnprob1 behsymp1;
read9grp on read1 lrnprob1 behsymp1 (beta1-beta3);
read2 on read9grp read1 lrnprob1 behsymp1;
stanread7 on read2 read9grp read1 lrnprob1 behsymp1;
```

The main table of results expands to include summaries of the auxiliary variable regression models. However, these additional parameters can be ignored because they are not the substantive focus. The logistic model's estimates and standard errors change, which is expected when including auxiliary variables that have salient semipartial correlations with the incomplete variables.

2.6 R mdmb FIML Script

The `lavaan` package currently does not offer maximum likelihood estimation for models with incomplete categorical variables. The example instead uses the `mdmb` package. This package leverages a factored regression specification that links incomplete predictors to one another using a sequence of univariate regression models. The additional regression equations are as follows.

$$\begin{aligned} BEHSYMP_1 &= \gamma_{01} + \epsilon_1 \\ LRNPROB_1 &= \gamma_{02} + \gamma_{12}(BEHSYMP_1) + \epsilon_2 \\ READ_1 &= \gamma_{03} + \gamma_{13}(LRNPROB_1) + \gamma_{23}(BEHSYMP_1) + \epsilon_3 \end{aligned} \tag{4}$$

These equations essentially comprise a path model where first grade behavioral symptom ratings predict learning problems, and both variables then predict first grade reading scores.

The R input file for the analysis is `Ex2.R`. The example requires the `mdmb` package.

R Script Ex2.R

```
1    library(mbmb)
2    load('behaviorachievement.rda')
3
4    summary(behaviorachievement)
```

The mdmb package requires the user to specify 'nodes' for the missing values. These nodes are essentially a fixed list of plausible score values that span each variable's range. Specifying these values is necessary for the optimization algorithm, which uses an imputation-like algorithm called numerical integration. The summary function on line 4 generates a table displaying the observed values of the numeric variables. The summary table is as follows.

```
   stanread7         read2            read1           lrnprob1        behsymp1
Min.   :100.0   Min.   : 20.00   Min.   : 39.00   Min.   :31.00   Min.   :17.00
1st Qu.:228.0   1st Qu.: 83.00   1st Qu.: 74.00   1st Qu.:45.00   1st Qu.:41.00
Median :263.0   Median : 92.00   Median : 86.00   Median :51.00   Median :48.00
Mean   :264.5   Mean   : 93.74   Mean   : 86.81   Mean   :52.36   Mean   :49.47
3rd Qu.:314.0   3rd Qu.:108.00   3rd Qu.: 99.00   3rd Qu.:60.50   3rd Qu.:58.00
Max.   :399.0   Max.   :150.00   Max.   :153.00   Max.   :88.00   Max.   :92.00
NA's   :27      NA's   :13       NA's   :9        NA's   :3       NA's   :5
```

The next part of the code creates variables that contain vectors of plausible replacement scores (nodes, pseudo-imputations) that span the entire range of the distributions. The binary outcome has only two possible scores, so its node vector on line 7 consists of 0s and 1s. For continuous variables, specifying 20 to 40 nodes is usually sufficient. For example, nodes.read1 is a vector of plausible scores ranging from 30 to 160 in increments of two, and nodes.lrnprb1 is a sequence of scores between 20 and 100 in increments of two. To account for the possibility that the missing scores fall outside the observed range, the vectors specify values beyond the minimum and maximum scores from the data.

R Script Ex2.R, continued

```
5    nodes.stanread7 <- seq(80, 420, by = 5)
6    nodes.read2 <- seq(10, 160, by = 2)
```

```
7    nodes.read9grp <- c(0,1)
8    nodes.read1 <- seq(30, 160, by = 2)
9    nodes.lrnprob1 <- seq(20, 100, by = 2)
10   nodes.behsymp1 <- seq(10, 100, by = 2)
```

The next part of the script specifies a model for each analysis variable and auxiliary variable. The predictor variable regressions from Equation 4 are listed first, followed by the logistic model from Equation 2. The auxiliary variable regressions from Equation 3 are last. Each model object includes three arguments: the type of regression (linear or logistic), an equation, and the incomplete variable's vector of nodes or pseudo-imputations. Linear regressions are specified with `'model' = 'linreg'` parameter, and the binary logistic regression is specified using `'model' = 'logistic'`.

R Script Ex2.R, continued

```
11   model.behsymp1 <- list('model' = 'linreg',
12     'formula' = behsymp1 ~ 1, nodes = nodes.behsymp1)
13   model.lrnprob1 <- list('model' = 'linreg',
14     'formula' = lrnprob1 ~ behsymp1, nodes = nodes.lrnprob1)
15   model.read1 <- list('model' = 'linreg',
16     'formula' = read1 ~ lrnprob1 + behsymp1, nodes = nodes.read1)
17   model.read9grp <- list('model' = 'logistic',
18     'formula' = read9grp ~ read1 + lrnprob1 + behsymp1,
19     nodes = nodes.read9grp)
20   model.read2 <- list('model' = 'linreg',
21     'formula' = read2 ~ read9grp + read1 + lrnprob1 + behsymp1,
22     nodes = nodes.read2)
23   model.stanread7 <- list('model' = 'linreg',
24     'formula' = stanread7 ~ read2 + read9grp + read1 + lrnprob1 +
25     behsymp1, nodes = nodes.stanread7)
```

The `mdmb` package views `stanread7` (the auxiliary variable in the final regression model) as the ultimate 'dependent' variable in the sequence, and it considers all other variables 'independent variables'. Starting on line 26, the final part of the code combines the independent variable models into a list. On lines 29 and 30, the data frame and the predictor list are passed

into the frm_em function, which fits the sequence of models. Finally, the summary function on line 31 requests tables of parameter estimates.

R Script Ex2.R, continued

```
26  predictor.models <- list(behsymp1 = model.behsymp1,
27    lrnprob1 = model.lrnprob1, read1 = model.read1,
28    read9grp = model.read9grp, read2 = model.read2)
29  fit <- frm_em(dat = behaviorachievement, dep = model.stanread7,
30    ind = predictor.models)
31  summary(fit)
```

2.7 R Output

The mdmb output includes a table of results for every fitted regression model. In this example, the output tables summarize linear regressions for the three incomplete predictors, a logistic regression for the binary dependent variable, and a pair of linear regressions for the auxiliary variables. These supporting model parameters are not of substantive interest, and they do not need to be reported. The output below shows the parameter estimates from the focal logistic model. The first two columns display the unstandardized estimates and their standard errors, the third and fourth columns display the corresponding *t*-statistics and *p*-values, and the rightmost columns contain 95% confidence interval limits.

```
Model 4: mdmb::logistic_regression( read9grp ~ read1 + lrnprob1 + behsymp1 )

  index       dv                    parm ON     est     se       t      p lower95 upper95
1    14 read9grp read9grp ON (Intercept)  1 -3.9045 1.6291 -2.3968 0.0165 -7.0974 -0.7117
2    15 read9grp       read9grp ON read1  1  0.0675 0.0149  4.5252 0.0000  0.0383  0.0968
3    16 read9grp    read9grp ON lrnprob1  1 -0.0225 0.0308 -0.7330 0.4636 -0.0828  0.0377
4    17 read9grp    read9grp ON behsymp1  1 -0.0192 0.0251 -0.7664 0.4434 -0.0685  0.0300

Pseudo R^2 (McKelvey & Zavoina)=0.4944
```

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 differ

by 0.07. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 4.53$, $p < .001$).

3

FIML: Regression With Binary and Ordinal Predictors

This example illustrates a multiple regression analysis with incomplete categorical predictors. The analysis uses the `mathachievement.dat` data set taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
MATHPOST	Math achievement posttest	18.0	Numeric
CONDITION	Experimental condition	0	0 = Comparison, 1 = Intervention
FRLUNCH	Lunch assistance code	4.4	0 = None, 1 = Free/reduced lunch
EFFICACY	Math self-efficacy rating	9.6	Ordinal (1 to 6)
MATHPRE	Math achievement pretest	0	Numeric
	Auxiliary Variables		
ATRISK	Behavioral disorder risk	5.2	0 = Low risk, 1 = At-risk
STANREAD	Standardized reading	9.2	Numeric

3.1 Analysis Model

The analysis model features math posttest scores regressed on the experimental condition and lunch assistance dummy codes, math self-efficacy ratings, and math pretest scores.

$$MATHPOST = \beta_0 + \beta_1(CONDITION) + \beta_2(FRLUNCH) + \beta_3(EFFICACY) + \beta_4(MATHPRE) + \varepsilon \quad (5)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. In this case, the predictor set includes incomplete binary and ordinal variables, so assigning a normal distribution to the variables is questionable.

The analysis instead uses a factored regression specification that separates the likelihood function into separate components for each variable type. In practical terms, this specification uses a sequence of univariate regression models to link incomplete predictors. The additional regression equations are logistic and linear models.

$$\begin{aligned} \text{logit}(FRLUNCH) &= \gamma_{01} + \gamma_{11}(CONDITION) + \gamma_{21}(MATHPRE) \\ EFFICACY &= \gamma_{02} + \gamma_{12}(FRLUNCH) + \gamma_{22}(CONDITION) + \gamma_{32}(MATHPRE) + \epsilon_2 \end{aligned} \quad (6)$$

These equations essentially comprise a path model where the intervention indicator and math pretest scores predict the lunch assistance indicator, and all three variables, in turn, predict self-efficacy. The two complete variables are always on the right side of regression equations because they do not require a model.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following Section 2.4, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} \text{logit}(ATRISK) &= \gamma_{03} + \gamma_{13}(MATHPOST) + \gamma_{23}(CONDITION) \\ &\quad + \gamma_{33}(FRLUNCH) + \gamma_{43}(EFFICACY) + \gamma_{53}(MATHPRE) \\ STANREAD &= \gamma_{04} + \gamma_{14}(ATRISK) + \gamma_{24}(MATHPOST) + \gamma_{34}(CONDITION) \\ &\quad + \gamma_{44}(FRLUNCH) + \gamma_{54}(EFFICACY) + \gamma_{64}(MATHPRE) + \epsilon_4 \end{aligned} \quad (7)$$

Again, the entire collection of regression equations can be viewed as a path model (see the auxiliary variable path diagram from Section 2.4). The key difference is that the path coefficients

are just a tool for linking variables with different metrics and do not represent a substantive theory.

3.2 Mplus FIML Script

The code block below shows Mplus script `Ex3.inp`.

Mplus Script Ex3.inp

```
1   DATA:
2   file = mathachievement.dat;
3   VARIABLE:
4   names = id condition male frlunch atrisk
5     stanread efficacy anxiety mathpre mathpost;
6   usevariables = mathpost condition frlunch efficacy
7     mathpre atrisk stanread;
8   categorical = frlunch efficacy atrisk;
9   missing = all(999);
10  ANALYSIS:
11  estimator = ml;
12  link = logit;
13  integration = montecarlo;
14  MODEL:
15  frlunch on condition mathpre;
16  efficacy on frlunch condition mathpre;
17  mathpost on condition frlunch efficacy mathpre (beta1-beta4);
18  atrisk on mathpost condition frlunch efficacy mathpre;
19  stanread on atrisk mathpost condition frlunch efficacy mathpre;
20  MODEL TEST:
21  0 = beta1; 0 = beta2; 0 = beta3; 0 = beta4;
22  OUTPUT:
23  patterns sampstat cinterval;
```

The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. The `VARIABLE` command provides information about the data. Beginning on line 4, the `names` subcommand assigns names to the variables in the input data file, the `usevariables` subcommand selects variables for the analysis, and the `missing` subcommand gives the global missing value code. The `categorical`

subcommand on line 8 defines three variables as either binary or ordinal. The `ANALYSIS` command and `estimator` subcommand specify full information maximum likelihood estimation. Finally, the `link = logit` option specifies a logistic regression for the outcome variable, and `integration = montecarlo` invokes an algorithmic method for models with mixed variable types (and a factored regression specification for the likelihood).

The `MODEL` section of the script consists of five lines. Lines 15 and 16 are logistic regressions linking the discrete predictors to the complete variables and each other (see Equation 6), and line 17 is the focal regression model from Equation 5. The end of this line includes parameter labels in parentheses. Finally, lines 18 and 19 are the auxiliary variable regressions shown in Equation 7. As noted previously, the collection of regressions can be viewed as a path model, with the focal regression as one part of a larger network (see the path diagram from Section 2.4). Next, the `MODEL TEST` command uses the labels from line 17 to specify a custom significance test of the null hypothesis that all three population slopes equal zero. Finally, the `OUTPUT` command specifies three keywords on line 23 that request a summary of the missing data patterns, maximum likelihood estimates of sample statistics, and confidence intervals (standardized coefficients are not available for this analysis).

3.3 Mplus Output

Information about the missing data patterns is found near the top of the output file. Following the missing data pattern table, the output displays a covariance coverage matrix that gives the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. The format of these tables is the same as those shown in Section 1.3.

The `MODEL TEST` command in the previous script requested an analogous Wald chi-square statistic that evaluates the null hypothesis that all population slopes equal zero. The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       31

...
```

Wald Test of Parameter Constraints

Value	149.182
Degrees of Freedom	4
P-Value	0.0000

The table of unstandardized parameter estimates is shown below. Because the analysis specifies a multivariate normal distribution for the predictors, the means, variances, and covariances of these variables are printed along with the focal model estimates. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown in bold typeface.

MODEL RESULTS

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value
FRLUNCH ON				
CONDITION	0.011	0.265	0.041	0.967
MATHPRE	-0.020	0.015	-1.290	0.197
EFFICACY ON				
FRLUNCH	-0.031	0.246	-0.125	0.901
CONDITION	0.506	0.240	2.107	0.035
MATHPRE	0.056	0.014	3.881	0.000
MATHPOST ON				
CONDITION	**2.306**	**1.023**	**2.255**	**0.024**
FRLUNCH	**-5.498**	**1.063**	**-5.173**	**0.000**
EFFICACY	**0.833**	**0.340**	**2.448**	**0.014**
MATHPRE	**0.526**	**0.061**	**8.594**	**0.000**
ATRISK ON				
MATHPOST	-0.028	0.025	-1.141	0.254
CONDITION	-0.080	0.342	-0.233	0.815
FRLUNCH	0.898	0.399	2.248	0.025

```
   EFFICACY          -0.337      0.115     -2.925      0.003
   MATHPRE           -0.018      0.024     -0.722      0.470

STANREAD   ON
   ATRISK           -13.492      1.231    -10.957      0.000
   MATHPOST           0.349      0.078      4.466      0.000
   CONDITION          1.493      1.019      1.466      0.143
   FRLUNCH           -2.435      1.177     -2.068      0.039
   EFFICACY          -0.478      0.351     -1.361      0.173
   MATHPRE            0.006      0.073      0.076      0.939

Intercepts
   MATHPOST          29.375      3.016      9.739      0.000
   STANREAD          44.135      4.035     10.938      0.000

Thresholds
   FRLUNCH$1         -0.623      0.780     -0.799      0.425
   EFFICACY$1         1.308      0.748      1.748      0.080
   EFFICACY$2         2.213      0.752      2.942      0.003
   EFFICACY$3         3.250      0.770      4.222      0.000
   EFFICACY$4         4.186      0.786      5.324      0.000
   EFFICACY$5         4.976      0.800      6.217      0.000
   ATRISK$1          -4.351      1.317     -3.304      0.001

Residual Variances
   MATHPOST          51.270      5.185      9.888      0.000
   STANREAD          52.261      5.226     10.000      0.000
```

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, consider the slope for the treatment assignment dummy code. The positive coefficient indicates that, for two students who share the same covariate profile (i.e., lunch assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.31 points higher than the student in the control group. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 2.26$, $p = .02$).

3.4 R mdmb FIML Script

The R input file for the analysis is `Ex3.R`. The example requires the `mdmb` package.

R Script Ex3.R

```
1    library(mbmb)
2    load('mathachievement.rda')
3
4    summary(mathachievement)
```

The mdmb package requires the user to specify 'nodes' for the missing values. These nodes are essentially a fixed list of plausible score values that span each variable's range. Specifying these values is necessary for the optimization algorithm, which uses an imputation-like algorithm called numerical integration. The summary function on line 4 generates a table displaying the observed values of the numeric variables. The summary table is as follows.

```
    frlunch        efficacy        mathpost         atrisk          stanread
 Min.   :0.00   Min.   :1.000   Min.   :37.00   Min.   :0.0000   Min.   :27.00
 1st Qu.:0.00   1st Qu.:2.000   1st Qu.:52.00   1st Qu.:1.0000   1st Qu.:45.00
 Median :0.00   Median :3.000   Median :57.00   Median :1.0000   Median :55.00
 Mean   :0.41   Mean   :3.394   Mean   :57.45   Mean   :0.7637   Mean   :52.52
 3rd Qu.:1.00   3rd Qu.:5.000   3rd Qu.:63.00   3rd Qu.:1.0000   3rd Qu.:60.50
 Max.   :1.00   Max.   :6.000   Max.   :85.00   Max.   :1.0000   Max.   :69.00
 NA's   :11     NA's   :24      NA's   :45      NA's   :13       NA's   :23
```

The next part of the code creates variables that contain vectors of plausible replacement scores (nodes, pseudo-imputations) that span the entire range of the distributions. The binary variables have only two possible scores, so their node vectors on lines 8 and 11 consist of 0s and 1s. On line 9, the efficacy scores similarly use integer nodes between 1 and 6. For continuous variables, specifying 20 to 40 nodes is usually sufficient. For example, nodes.stanread is a vector of plausible scores ranging from 20 to 80 in increments of two, and nodes.mathpost is a sequence of scores between 30 and 90 in increments of two. To account for the possibility that the missing scores fall outside the observed range, the vectors specify values beyond the minimum and maximum scores from the data.

R Script Ex3.R, continued

```
5    nodes.frlunch <- c(0,1)
6    nodes.efficacy <- seq(1, 6, by = 1)
7    nodes.mathpost <- seq(30, 90, by = 2)
8    nodes.atrisk <- c(0,1)
9    nodes.stanread <- c(20, 80, by = 2)
```

The next part of the script specifies a model for each analysis variable and auxiliary variable. The predictor variable regressions from Equation 6 are listed first, followed by the focal model from Equation 5. The auxiliary variable regressions from Equation 7 are last. Each model object includes three arguments: the type of regression (linear or logistic), an equation, and the incomplete variable's nodes. Linear regressions are specified with `'model' = 'linreg'` parameter, and the binary logistic regression is specified using `'model' = 'logistic'`.

R Script Ex3.R, continued

```
10   model.frlunch <- list('model' = 'logistic',
11     'formula' = frlunch ~ condition + mathpre,
12     nodes = nodes.frlunch)
13   model.efficacy <- list('model' = 'linreg',
14     'formula' = efficacy ~ frlunch + condition + mathpre,
15     nodes = nodes.efficacy)
16   model.mathpost <- list('model' = 'linreg',
17     'formula' = mathpost ~ condition + frlunch + efficacy + mathpre,
18     nodes = nodes.mathpost)
19   model.atrisk <- list('model' = 'logistic',
20     'formula' = atrisk ~ mathpost + condition + frlunch + efficacy +
21     mathpre, nodes = nodes.atrisk)
22   model.stanread <- list('model' = 'linreg',
23     'formula' = stanread ~ atrisk + mathpost + condition + frlunch +
24     efficacy + mathpre, nodes = nodes.stanread)
```

The `mdmb` package views `stanread` (the auxiliary variable in the final regression model) as the ultimate 'dependent' variable, and it considers all other variables 'independent variables'. Starting on line 25, the final part of the code combines the independent variable models into a list. On lines 28 and 29, the data frame and the predictor list are passed into the `frm_em` function, which

fits the sequence of models. Finally, the summary function on line 30 requests tables of parameter estimates.

R Script Ex3.R, continued

```
25  predictor.models <- list(frlunch = model.frlunch,
26    efficacy = model.efficacy, mathpost = model.mathpost,
27    atrisk = model.atrisk)
28  fit <- frm_em(dat = mathachievement, dep = model.stanread,
29    ind = predictor.models)
30  summary(fit)
```

3.5 R Output

The mdmb output includes a table of results for every fitted regression model. The supporting model parameters are not of substantive interest, and they do not need to be reported. The output below shows the parameter estimates from the focal regression model. The first two columns display the unstandardized estimates and their standard errors, the third and fourth columns display the corresponding *t*-statistics and *p*-values, and the rightmost columns contain 95% confidence interval limits.

```
Model 3: stats::lm( mathpost ~ condition + frlunch + efficacy + mathpre )

  index       dv                    parm ON     est     se       t      p lower95 upper95
1    15 mathpost mathpost ON (Intercept)  1 29.0504 3.0085  9.6562 0.0000 23.1539 34.9469
2    16 mathpost   mathpost ON condition  1  2.2939 1.0226  2.2431 0.0249  0.2895  4.2982
3    17 mathpost     mathpost ON frlunch  1 -5.2352 1.0592 -4.9427 0.0000 -7.3111 -3.1593
4    18 mathpost    mathpost ON efficacy  1  0.7966 0.3391  2.3490 0.0188  0.1319  1.4612
5    19 mathpost     mathpost ON mathpre  1  0.5200 0.0607  8.5687 0.0000  0.4011  0.6390
6    20 mathpost          mathpost sigma  0  7.1076 0.3524 20.1691 0.0000  6.4169  7.7983

Explained variance R^2=0.4197
```

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, the positive coefficient for the treatment assignment predictor indicates that, for two students who share the same covariate profile (i.e., lunch

assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.29 points higher than the student in the control group. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 2.24$, $p = .03$).

4

FIML: Moderated Regression With an Interaction

This example illustrates a multiple regression analysis with an incomplete interaction effect. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
ATRISK	Emotion/behavior disorder risk	2.2	0 = No risk, 1 = At risk
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

4.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement, teacher-rated learning problems, and the product of first grade reading scores and learning problems, and a binary risk indicator.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(READ_1)(LRNPROB_1) + \beta_4(ATRISK) + \varepsilon \quad (8)$$

Moderated regression models (and models with non-linearities more generally) require a factored regression specification that splits the likelihood into separate parts for the outcome model and predictors.

Unlike a complete-data regression analysis, incomplete variables also require distributional assumptions and models that define those distributions. The analysis uses a factored regression specification that separates the likelihood function into separate components for each variable. The analysis uses a logistic regression for the binary covariate and linear models for the other two predictors. In practical terms, this specification uses a sequence of univariate regression models to link incomplete predictors. The additional regression equations are as follows.

$$\begin{aligned} \text{logit}(ATRISK) &= \gamma_{01} \\ LRNPROB_1 &= \gamma_{02} + \gamma_{12}(ATRISK) + \epsilon_2 \\ READ_1 &= \gamma_{03} + \gamma_{13}(LRNPROB_1) + \gamma_{23}(BEHSYMP_1) + \epsilon_3 \end{aligned} \quad (9)$$

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} READ_2 &= \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \gamma_{31}(LRNPROB_1) + \gamma_{41}(ATRISK) + \epsilon_4 \\ STANREAD_7 &= \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \gamma_{32}(READ_1) \\ &\quad + \gamma_{42}(LRNPROB_1) + \gamma_{52}(ATRISK) + \epsilon_5 \end{aligned} \quad (10)$$

Along with the other models, the collection of regression equations can be viewed as a path model where the focal analysis is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

4.2 R mdmb FIML Script

The example uses the `lavaan` and `mdmb` packages. The latter leverages a factored regression specification that links incomplete predictors to one another using a sequence of univariate regression models. R input file for the analysis is `Ex4.R`. The code block below shows the commands that import and modify the data.

R Script Ex4.R

```
1   library(lavaan)
2   library(mbmb)
3   load('behaviorachievement.rda')
4
5   model <- 'stanread7 ~ 1; read2 ~ 1; read9 ~ 1; read1 ~ 1;
6     lrnprob1 ~ 1; atrisk ~ 1;'
7   descriptives <- inspectSampleCov(model, behaviorachievement,
8     missing = 'fiml')
9
10  behaviorachievement$read1.cgm <-
11    behaviorachievement$read1 - descriptives$mean['read1']
12  behaviorachievement$lrnprob1.cgm <-
13    data$lrnprob1 - descriptives$mean['lrnprob1']
14
15  summary(behaviorachievement)
```

The analysis centers the two predictors involved in the interaction at their grand means. Because the predictors are incomplete, the script uses `lavaan` to obtain maximum likelihood-estimated means for centering. The `model` variable on lines 5 and 6 defines a text string describing a set of empty regression models with only an intercept (the `~ 1` after each variable name). Along with the data frame, this model is passed into `lavaan`'s `inspectSampleCov` function on line 7. The resulting maximum likelihood estimates of the means, which are stored in the object called `descriptives`, are used to create new centered variables called `read1.cgm` and `lrnprob1.cgm` beginning on line 10.

The `mdmb` package requires the user to specify 'nodes' for the missing values. These nodes are essentially a fixed list of plausible score values that span each variable's range. Specifying these values is necessary for the optimization algorithm, which uses an imputation-like algorithm

called numerical integration. The `summary` function on line 15 generates a table displaying the observed values of the numeric variables. The summary table is as follows.

```
   stanread7         read2           read9          read1.cgm         lrnprob1.cgm          atrisk
Min.   :100.0   Min.   : 20.00   Min.   : 41.00   Min.   :-47.1819   Min.   :-21.34409   Min.   :0.0000
1st Qu.:228.0   1st Qu.: 83.00   1st Qu.: 81.00   1st Qu.:-12.1819   1st Qu.: -7.34409   1st Qu.:0.0000
Median :263.0   Median : 92.00   Median : 89.00   Median : -0.1819   Median : -1.34409   Median :1.0000
Mean   :264.5   Mean   : 93.74   Mean   : 88.55   Mean   :  0.6243   Mean   :  0.01147   Mean   :0.6519
3rd Qu.:314.0   3rd Qu.:108.00   3rd Qu.: 97.00   3rd Qu.: 12.8181   3rd Qu.:  8.15591   3rd Qu.:1.0000
Max.   :399.0   Max.   :150.00   Max.   :123.00   Max.   : 66.8181   Max.   : 35.65591   Max.   :1.0000
NA's   :27      NA's   :13       NA's   :24       NA's   :9          NA's   :3           NA's   :3
```

The next part of the code creates variables that contain vectors of plausible replacement scores that span the entire range of the distributions. For continuous variables, specifying 20 to 40 nodes is usually sufficient. For example, `nodes.read1` is a vector of plausible centered scores ranging from –55 to 75 in increments of two, and `nodes.lrnprb1` is a sequence of centered scores between –30 and 50 in increments of two. To account for the possibility that the missing scores fall outside the observed range, the vectors specify values beyond the minimum and maximum scores from the data. The binary predictor has only two node values.

R Script Ex4.R, continued

```
16  nodes.stanread7 <- seq(80, 420, by = 5)
17  nodes.read2 <- seq(10, 160, by = 5)
18  nodes.read9 <- seq(30, 130, by = 2)
19  nodes.read1 <- seq(-55, 75, by = 2)
20  nodes.lrnprob1 <- seq(-30, 50, by = 2)
21  nodes.atrisk <- c(0,1)
```

The next part of the script specifies a model for each analysis variable and auxiliary variable. The predictor variable regressions from Equation 9 are listed first, followed by the focal moderated regression model from Equation 8. The auxiliary variable regressions from Equation 10 are last. Each model object includes three arguments: the type of regression (linear or logistic), an equation, and the incomplete variable's vector of nodes or pseudo-imputations. Note that the focal model list beginning on line 31 includes the product of two centered variables.

R Script Ex4.R, continued

```
22  model.behsymp1 <- list( 'model' = 'logistic',
23    'formula' = atrisk ~ 1,
24    nodes = nodes.atrisk)
25  model.lrnprob1 <- list( 'model' = 'linreg',
26    'formula' = lrnprob1.cgm ~ behsymp1,
27    nodes = nodes.lrnprob1)
28  model.read1 <- list( 'model' = 'linreg',
29    'formula' = read1.cgm ~ lrnprob1.cgm + behsymp1,
30    nodes = nodes.read1)
31  model.read9 <- list( 'model' = 'linreg',
32    'formula' = read9 ~ read1.cgm + lrnprob1.cgm +
33    read1.cgm*lrnprob1.cgm + behsymp1,
34    nodes = nodes.read9)
35  model.read2 <- list('model' = 'linreg',
36    'formula' = read2 ~ read9 + read1.cgm + lrnprob1.cgm + behsymp1,
37    nodes = nodes.read2)
38  model.stanread7 <- list('model' = 'linreg',
39    'formula' = stanread7 ~ read2 + read9 + read1.cgm
40    + lrnprob1.cgm + behsymp1, nodes = nodes.stanread7)
```

The `mdmb` package views `stanread7` (the auxiliary variable in the final regression model) as the ultimate 'dependent' variable in the sequence, and it considers all other variables as 'independent variables'. Starting on line 41, the final part of the code combines the independent variable models into a list. On line 44, the data frame and the predictor list are passed into the `frm_em` function, which fits the sequence of models. Finally, the `summary` function on line 46 requests tables of parameter estimates.

R Script Ex4.R, continued

```
41  predictor.models <- list(atrisk = model.atrisk,
42    lrnprob1 = model.lrnprob1, read1 = model.read1, read9 = model.read9,
43    read2 = model.read2)
44  fit <- frm_em(dat = behaviorachievement, dep = model.stanread7,
45    ind = predictor.models)
46  summary(fit)
```

4.3 R Output

The `mdmb` output includes a table of results for every fitted regression model. The supporting model parameters are not of substantive interest, and they do not need to be reported. The output below shows the parameter estimates from the focal model. The first two columns display the unstandardized estimates and their standard errors, the third and fourth columns display the corresponding *t*-statistics and *p*-values, and the rightmost columns contain 95% confidence interval limits.

```
Model 4: stats::lm( read9 ~ read1.cgm + lrnprob1.cgm + read1.cgm * lrnprob1.cgm + atrisk )

  index    dv                              parm ON     est     se       t      p lower95 upper95
1    14 read9             read9 ON (Intercept)  1 89.0374 1.4195 62.7261 0.0000 86.2553 91.8195
2    15 read9               read9 ON read1.cgm  1  0.5053 0.0437 11.5510 0.0000  0.4195  0.5910
3    16 read9            read9 ON lrnprob1.cgm  1 -0.3785 0.0833 -4.5451 0.0000 -0.5417 -0.2153
4    17 read9                  read9 ON atrisk  1 -1.9092 1.7952 -1.0635 0.2875 -5.4278  1.6093
5    18 read9 read9 ON read1.cgm:lrnprob1.cgm  1  0.0128 0.0045  2.8247 0.0047  0.0039  0.0216
6    19 read9                      read9 sigma  0  9.1591 0.6412 14.2841 0.0000  7.9023 10.4158

Explained variance R^2=0.6183
```

The lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of first grade reading scores ($\hat{\beta}_1 = 0.51$) is the effect of that predictor at the mean of the first-grade learning problems, and the learning problems slope ($\hat{\beta}_2 = -0.38$) similarly reflects a conditional effect at the reading score mean. The interaction slope captures the change in the first-grade reading slope for each one-unit increase in learning problems (and vice versa). Specifically, the positive coefficient ($\hat{\beta}_3 = 0.013$) indicates that the association between first and ninth grade reading scores becomes stronger (i.e., more positive) as learning problems increase. That is, the predictive power of early reading on later reading is strongest for students with elevated learning problem ratings in first grade.

5

FIML: Curvilinear Regression

This example illustrates a multiple regression analysis with an incomplete curvilinear effect. The analysis uses the mathachievement.dat data set taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
MATHPOST	Math achievement posttest	18.0	Numeric
ANXIETY	Math anxiety composite	8.4	Numeric
FRLUNCH	Lunch assistance code	4.4	0 = None, 1 = Free/reduced lunch
EFFICACY	Math self-efficacy rating	9.6	Ordinal (1 to 6)
MATHPRE	Math achievement pretest	0	Numeric
	Auxiliary Variables		
ATRISK	Behavioral disorder risk	5.2	0 = Low risk, 1 = At-risk
STANREAD	Standardized reading	9.2	Numeric

5.1 Analysis Model

The analysis model features math posttest scores regressed on anxiety and its square, the lunch assistance dummy code, math self-efficacy ratings, and math pretest scores.

$$MATHPOST = \beta_0 + \beta_1(ANXIETY) + \beta_2(ANXIETY^2) + \beta_3(FRLUNCH) + \beta_4(EFFICACY) + \beta_5(MATHPRE) + \varepsilon \quad (11)$$

Curvilinear regression models (and models with non-linearities more generally) require a factored regression specification that splits the likelihood into separate parts for the outcome model and predictors.

Unlike a complete-data regression analysis, incomplete variables also require distributional assumptions and models that define those distributions. The analysis uses a factored regression specification that separates the likelihood function into separate components for each variable. In practical terms, this specification uses a sequence of univariate regression models to link incomplete predictors. The additional regression equations, one of which is a logistic model, are as follows.

$$\begin{aligned} \text{logit}(FRLUNCH) &= \gamma_{01} + \gamma_{11}(MATHPRE) \\ EFFICACY &= \gamma_{02} + \gamma_{12}(FRLUNCH) + \gamma_{22}(MATHPRE) + \epsilon_2 \\ ANXIETY &= \gamma_{03} + \gamma_{13}(EFFICACY) + \gamma_{23}(FRLUNCH) + \gamma_{33}(MATHPRE) + \epsilon_3 \end{aligned} \quad (12)$$

These equations essentially comprise a path model where math pretest scores predict the lunch assistance indicator, the lunch assistant dummy code and math pretest scores predict efficacy, and all three variables, in turn, predict anxiety. The complete variable is always on the right side of regression equations because it does not require a model.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} \text{logit}(ATRISK) &= \gamma_{04} + \gamma_{14}(MATHPOST) + \gamma_{24}(ANXIETY) + \gamma_{34}(FRLUNCH) + \gamma_{44}(EFFICACY) + \gamma_{54}(MATHPRE) \\ STANREAD &= \gamma_{05} + \gamma_{15}(ATRISK) + \gamma_{25}(MATHPOST) + \gamma_{35}(ANXIETY) + \gamma_{45}(FRLUNCH) + \gamma_{55}(EFFICACY) + \gamma_{65}(MATHPRE) + \epsilon_5 \end{aligned} \quad (13)$$

Again, the entire collection of regression equations can be viewed as a path model where the curvilinear regression is one piece of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

5.2 R mdmb FIML Script

The example uses the `lavaan` and `mdmb` packages. The latter leverages a factored regression specification that links incomplete predictors to one another using a sequence of univariate regression models. R input file for the analysis is `Ex5.R`. The code block below shows the commands that import and modify the data.

R Script Ex5.R

```
1   Library(lavaan)
2   library(mbmb)
3   load('mathachievement.rda')
4
5   model <- 'stanread ~ 1; atrisk ~ 1; mathpost ~ 1; anxiety ~ 1;
6     frlunch ~ 1; efficacy ~ 1; mathpre ~ 1;'
7   descriptives <- inspectSampleCov(model, mathachievement,
8     missing = 'fiml')
9
10  mathachievement$anxiety.cgm <-
11    mathachievement$anxiety - descriptives$mean['anxiety']
12
13  summary(mathachievement)
```

The analysis centers math anxiety (the curvilinear predictor) at its grand mean. Because the predictors are incomplete, the script uses `lavaan` to obtain maximum likelihood-estimated means for centering. The `model` variable on lines 5 and 6 defines a text string describing a set of empty regression models with only an intercept (the `~ 1` after each variable name). Along with the data frame, this model is passed into `lavaan`'s `inspectSampleCov` function on line 7. The resulting maximum likelihood estimates of the means, which are stored in the object called `descriptives`, are used to create new centered variables called `read1.cgm` and `lrnprob1.cgm` beginning on line 10.

The mdmb package requires the user to specify 'nodes' for the missing values. These nodes are essentially a fixed list of plausible score values that span each variable's range. Specifying these values is necessary for the optimization algorithm, which uses an imputation-like algorithm called numerical integration. The summary function on line 13 generates a table displaying the observed values of the numeric variables. The summary table is as follows.

```
    stanread        mathpost      anxiety.cgm          efficacy        frlunch         atrisk
 Min.   :27.00   Min.   :37.00   Min.   :-18.2628   Min.   :1.000   Min.   :0.00   Min.   :0.0000
 1st Qu.:45.00   1st Qu.:52.00   1st Qu.: -5.2628   1st Qu.:2.000   1st Qu.:0.00   1st Qu.:1.0000
 Median :55.00   Median :57.00   Median : -1.2628   Median :3.000   Median :0.00   Median :1.0000
 Mean   :52.52   Mean   :57.45   Mean   : -0.1056   Mean   :3.394   Mean   :0.41   Mean   :0.7637
 3rd Qu.:60.50   3rd Qu.:63.00   3rd Qu.:  3.7372   3rd Qu.:5.000   3rd Qu.:1.00   3rd Qu.:1.0000
 Max.   :69.00   Max.   :85.00   Max.   : 25.7372   Max.   :6.000   Max.   :1.00   Max.   :1.0000
 NA's   :23      NA's   :45      NA's   :21         NA's   :24      NA's   :11     NA's   :13
```

The next part of the code creates variables that contain vectors of plausible replacement scores (nodes, pseudo-imputations) that span the entire range of the distributions. For continuous variables, specifying 20 to 40 nodes is usually sufficient. For example, nodes.mathpost is a sequence of raw scores between 30 and 90 in increments of two, and nodes.anxiety is a vector of plausible centered scores ranging from –30 to 30 in increments of two. To account for the possibility that the missing scores fall outside the observed range, the vectors specify values beyond the minimum and maximum scores from the data.

R Script Ex5.R, continued

```
14  nodes.frlunch <- c(0,1)
15  nodes.efficacy <- seq(1, 6, by = 1)
16  nodes.mathpost <- seq(30, 90, by = 2)
17  nodes.anxiety <- seq(-30, 30, by = 2)
18  nodes.atrisk <- c(0,1)
19  nodes.stanread <- c(20, 80, by = 2)
```

The next part of the script specifies a model for each analysis variable and auxiliary variable. The predictor variable regressions from Equation 12 are listed first, followed by the focal model from Equation 11. The auxiliary variable regressions from Equation 13 are last. Each model object includes three arguments: the type of regression (linear or logistic), an equation, and the incomplete variable's vector of nodes or pseudo-imputations. Linear regressions are specified

with `'model' = 'linreg'` parameter, and the binary logistic regression is specified using `'model' = 'logistic'`. Note that the focal model list beginning on line 29 includes the square of the centered variable (i.e., `I(anxiety.cgm^2)`).

R Script Ex5.R, continued

```
20  model.frlunch <- list('model' = 'logistic',
21    'formula' = frlunch ~ mathpre,
22    nodes = nodes.frlunch)
23  model.efficacy <- list('model' = 'linreg',
24    'formula' = efficacy ~ frlunch + mathpre,
25    nodes = nodes.efficacy)
26  model.anxiety <- list('model' = 'linreg',
27    'formula' = anxiety ~ efficacy + frlunch + mathpre,
28    nodes = nodes.anxiety)
29  model.mathpost <- list('model' = 'linreg',
30    'formula' = mathpost ~ anxiety.cgm + I(anxiety.cgm^2) +
31    frlunch + efficacy + mathpre, nodes = nodes.mathpost)
32  model.atrisk <- list('model' = 'logistic',
33    'formula' = atrisk ~ mathpost + anxiety + frlunch + efficacy +
34    mathpre, nodes = nodes.atrisk)
35  model.stanread <- list('model' = 'linreg',
36    'formula' = stanread ~ atrisk + mathpost + anxiety + frlunch +
37    efficacy + mathpre, nodes = nodes.stanread)
```

The `mdmb` package views `stanread` (the auxiliary variable in the final regression model) as the ultimate 'dependent' variable in the sequence, and it considers all other variables 'independent variables'. Starting on line 38, the final part of the code combines the independent variable models into a list. On line 41, the data frame and the predictor list are passed into the `frm_em` function, which fits the sequence of models. Finally, the `summary` function on line 43 requests tables of parameter estimates.

R Script Ex5.1.R, continued

```
38  redictor.models <- list(frlunch = model.frlunch,
39    efficacy = model.efficacy, anxiety = model.anxiety,
40    mathpost = model.mathpost, atrisk = model.atrisk)
```

```
fit <- frm_em(dat = mathachievement, dep = model.stanread,
  ind = predictor.models)
summary(fit)
```

5.3 R Output

The mdmb output includes a table of results for every fitted regression model. The supporting model parameters are not of substantive interest, and they do not need to be reported. The output below shows the parameter estimates from the focal curvilinear model. The first two columns display the unstandardized estimates and their standard errors, the third and fourth columns display the corresponding *t*-statistics and *p*-values, and the rightmost columns contain 95% confidence interval limits.

```
Model 4: stats::lm( mathpost ~ anxiety.cgm + I(anxiety.cgm^2) + efficacy + frlunch + mathpre )

  index       dv                          parm ON     est     se       t      p lower95 upper95
1    15 mathpost      mathpost ON (Intercept)  1 33.2388 3.3678  9.8695 0.0000 26.6380 39.8396
2    16 mathpost      mathpost ON anxiety.cgm  1  0.0398 0.0793  0.5015 0.6160 -0.1156  0.1952
3    17 mathpost mathpost ON I(anxiety.cgm^2)  1 -0.0209 0.0059 -3.5452 0.0004 -0.0324 -0.0093
4    18 mathpost         mathpost ON efficacy  1  1.0629 0.3324  3.1975 0.0014  0.4114  1.7145
5    19 mathpost          mathpost ON frlunch  1 -5.5373 1.0398 -5.3255 0.0000 -7.5752 -3.4994
6    20 mathpost          mathpost ON mathpre  1  0.4648 0.0651  7.1361 0.0000  0.3371  0.5925
7    21 mathpost               mathpost sigma  0  6.9386 0.3460 20.0511 0.0000  6.2604  7.6168
```

In a curvilinear regression model, the lower-order term for math anxiety is a conditional effect that depends on scaling or centering. The slope conveys the instantaneous linear change in the outcome at the anxiety mean, controlling for all other predictors ($\hat{\beta}_1 = 0.04$). The negative quadratic coefficient ($\hat{\beta}_2 = -0.02$) indicates that the positive association at the mean decreases (i.e., becomes less positive) as anxiety increases (and vice versa). At high enough levels of anxiety, the association becomes negative, such that anxiety has a debilitating effect on math performance.

6

MCMC: Regression With Multivariate Normal Data

This example illustrates a multiple regression analysis with multivariate normal incomplete data. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

6.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) + \varepsilon \quad (14)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. By default, Blimp invokes a multivariate normal distribution for predictors.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following the same factored regression specification from earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} READ_2 = \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \ \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1) + \epsilon_1 \\ STANREAD_7 = \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \ \gamma_{32}(READ_1) \\ + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1) + \epsilon_2 \end{aligned} \quad (15)$$

Along with the focal regression model from Equation 14, the collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

6.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex6.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex6.1.imp

```
DATA: behaviorachievement.dat;
VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
   read1 read2 read3 read9 read9grp stanread7
   math1 math2 math3 math9 math9grp stanmath7;
MISSING: 999;
MODEL:
```

```
read9 ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
stanread7 read2 ~ read9 read1 lrnprob1 behsymp1;
WALDTEST: beta1:beta3 = 0;
SEED: 90291;
BURN: 1000;
ITERATIONS: 10000;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns, and the MISSING command on line 5 defines a global missing value code as 999.

The MODEL and WALDTEST blocks can be viewed as a set. The MODEL command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. Line 7 assigns labels the slope coefficients using the @ symbol. Blimp automatically configures the explanatory variable models under the assumption that they are normally distributed. Line 8 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 15; in the first model, READ2 is regressed on the focal variables, and the second model features STANREAD7 regressed on READ2 and the focal variables. The WALDTEST command uses the parameter labels to specify a custom hypothesis test that all three slopes equal zero. This so-called Bayesian Wald test (Asparouhov & Muthén, 2021) is a frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test.

Finally, lines 10 through 12 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

Blimp prints a table of regression results for each outcome variable to the left of a tilde, and it orders the tables alphabetically. In this example, the focal model's table would not appear first on the output. Blimp allows users to order tables by assigning labels to blocks of regression equations. To illustrate, the code block below assigns the label `focal.model` to main regression and the label `auxiliary.models` to the auxiliary variable regressions. Because output tables are listed in the same order as the labels, the focal results would now appear before the ancillary model results.

```
MODEL:
focal.model:
read9 ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
auxiliary.models:
stanread7 read2 ~ read9 read1 lrnprob1 behsymp1;
```

The corresponding `rblimp` script `Ex6.R` is shown below.

rblimp Script Ex6.R

```
library(rblimp)
load('behaviorachievement.rda')

mymodel <- rblimp(
  data = behaviorachievement,
  model = '
    focal.model:
    read9 ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
    auxiliary.models:
    stanread7 read2  ~ read9 read1 lrnprob1 behsymp1',
  waldtest = 'beta1:beta3 = 0',
  seed = 90291,
  burn = 1000,
  iter = 10000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp` function. The two exceptions are the `VARIABLES` and `MISSING` commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the `MODEL` command) are separated by semicolons, as they are in the Blimp script. Finally, the `output(mymodel)` function prints the Blimp output.

6.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           26 to 50              1.263            15
                           51 to 100             1.081            41
                           76 to 150             1.056            37
                          101 to 200             1.037            26
                          126 to 250             1.059            32
                          151 to 300             1.027            17
                          176 to 350             1.031            41
                          201 to 400             1.022            33
                          226 to 450             1.034            17
                          251 to 500             1.020            15
                          276 to 550             1.027            20
                          301 to 600             1.023            44
                          326 to 650             1.014            19
                          351 to 700             1.010            45
                          376 to 750             1.014            33
                          401 to 800             1.012            33
                          426 to 850             1.017            37
                          451 to 900             1.023            41
                          476 to 950             1.025            41
                          501 to 1000            1.016            41
```

The next output excerpt shows information about the variables in the analysis and the models used for estimation. The MODELS summary section is reserved for outcome variables that appear to the left of a tilde symbol. In this example, Blimp automatically constructs supporting models for incomplete predictor variables, so these models are omitted from the table.

```
DATA INFORMATION:

  Sample Size:              138
  Missing Data Rates:

                    read9 = 17.39
                    read2 = 09.42
                stanread7 = 19.57
                 behsymp1 = 03.62
                 lrnprob1 = 02.17
                    read1 = 06.52

MODEL INFORMATION:

  NUMBER OF PARAMETERS
    Outcome Models:         18
    Predictor Models:       12

  PREDICTORS
    Incomplete continuous:  behsymp1 lrnprob1 read1

MODELS

   focal.model:
    [1]  read9 ~ Intercept read1@beta1 lrnprob1@beta2 behsymp1@beta3

   auxiliary.models:
    [2]  read2 ~ Intercept read9 read1 lrnprob1 behsymp1
    [3]  stanread7 ~ Intercept read2 read9 read1 lrnprob1 behsymp1
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of

the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  read9

Parameters                    Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
Variances:
  Residual Var.               91.648     12.834     70.678    120.824        ---        ---   5905.634

Coefficients:
  Intercept                   66.011      6.144     54.152     78.192    115.461      0.000   6878.374
  read1                        0.504      0.044      0.419      0.590    131.211      0.000   7084.878
  lrnprob1                    -0.247      0.120     -0.479     -0.001      4.204      0.040   5865.510
  behsymp1                    -0.183      0.105     -0.389      0.025      2.994      0.084   6365.617

Standardized Coefficients:
  read1                        0.688      0.040      0.599      0.756    289.341      0.000   6613.583
  lrnprob1                    -0.178      0.085     -0.341     -0.001      4.265      0.039   5718.193
  behsymp1                    -0.147      0.084     -0.311      0.020      3.042      0.081   6399.987
```

```
Proportion Variance Explained
  by Coefficients                0.594      0.050      0.485      0.681        ---        ---   6288.304
  by Residual Variation          0.406      0.050      0.319      0.515        ---        ---   6288.304

                                 ------------------------------------------------------------------------------
```

To begin, the `N_Eff` values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the `N_Eff` values are insufficient, increasing the value on the `ITERATIONS` command will remedy the issue.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by 0.50 points on `READ9`. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. The standardized coefficients convey the expected change in standard deviation units for a one standard deviation increase in each predictor. For example, the model predicts that two individuals who differ by one standard deviation on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by 0.69 standard deviations on `READ9`. Collectively, the predictors explain 60% of the variation in ninth-grade reading scores. Note that the tabled values are numerically equivalent to the maximum likelihood estimates in Chapter 1.

The Blimp output also includes tables of regression model parameters for the auxiliary variables as well as the auto-generated models for incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. An example table is shown below. These additionally results are not of substantive interest and would not be reported.

```
PREDICTOR MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.
```

```
Missing predictor:  behsymp1

Parameters                      Median      StdDev       2.5%      97.5%        PSR      N_Eff
                              ------------------------------------------------------------------

Grand Mean                      49.518       1.066     47.435     51.607      1.000   3339.276

Level 1:
  lrnprob1                       0.799       0.070      0.661      0.933      1.000   8912.467
  read1                         -0.012       0.037     -0.083      0.061      1.000   8446.759
  Residual Var.                 73.066       9.256     57.795     94.382      1.000   7751.755

                              ------------------------------------------------------------------
```

Finally, recall that the `WALDTEST` command requested a Bayesian Wald chi-square statistic (Asparouhov & Muthén, 2021) that evaluates the null hypothesis that all population slopes equal zero. To reiterate, the Wald test is frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test. The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT` section under the `WALD TEST` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT:


  INFORMATION CRITERIA

    Marginal Likelihood
      DIC2                  3425.311
      WAIC                  3459.204

    Conditional Likelihood
      DIC2                  3425.311
      WAIC                  3459.204

  WALD TESTS (Asparouhov & Muthén, 2021)
```

```
Test #1

  Full:
    [1]  read9 ~ Intercept read1@beta1 lrnprob1@beta2 behsymp1@beta3

  Restricted:
    [1]  read9 ~ Intercept read1@beta1 lrnprob1@beta2 behsymp1@beta3

  Constraints in Restricted:
    [1]  beta1 = 0
    [2]  beta2 = 0
    [3]  beta3 = 0


  Wald Statistic (Chi-Square)              165.486
  Number of Parameters Tested (df)               3
  Probability                                0.000
```

6.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex6.2.imp` is identical `Ex6.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword

saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
    read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex6.R

```
1  library(rblimp)
2  load('behaviorachievement.rda')
3
```

```
mymodel <- rblimp(
  data = behaviorachievement,
  model = '
    focal.model:
    read9 ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
    auxiliary.models:
    stanread7 read2  ~ read9 read1 lrnprob1 behsymp1',
  waldtest = 'beta1:beta3 = 0',
  seed = 90291,
  burn = 1000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The SAVE command is no longer necessary because imputations are automatically stored in a rblimp object called mymodel@imputations. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

6.5 Analyzing Multiple Imputations in R

Continuing with the previous rblimp script, the following excerpt from Ex6.R shows how to perform multiple imputation inference. The script requires the mitml package (Grund et al., 2023).

R Script Ex6.R

```
library(rblimp)
library(mitml)
load('behaviorachievement.rda')

mymodel <- rblimp(...)

implist <- as.mitml(mymodel)
fit <- with(implist, lm(read9 ~ read1 + lrnprob1 + behsymp1))
estimates <- testEstimates(fit, extra.pars = T, df.com = 134)
estimates
```

```
21  confint(estimates)
22
23  null <- with(implist, lm(read9 ~ 1))
24  testModels(fit, null, df.com = 134, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Line 8 fits the focal regression model using the `lm` function, and line 9 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 10 and 11 print the pooled estimates and confidence intervals. Finally, lines 13 and 14 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 13 that represents the null hypothesis, which in this case is an empty regression model with just an intercept. On line 14, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

6.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    65.487     5.877    11.144   100.498     0.000     0.169     0.161
read1           0.506     0.043    11.725    92.752     0.000     0.212     0.192
lrnprob1       -0.231     0.114    -2.022   100.704     0.046     0.168     0.160
```

```
behsymp1       -0.189     0.102    -1.841    97.962     0.069     0.182     0.171

                   Estimate
Residual~~Residual   88.944

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.

                 2.5 %      97.5 %
(Intercept) 53.8288728 77.14584684
read1        0.4202903  0.59168880
lrnprob1    -0.4581615 -0.00433096
behsymp1    -0.3919669  0.01475078
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by .51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 11.73$, $p < .001$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation. Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     58.272       3 123.487   0.000   0.177

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.
```

6.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data

set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). The contents of this file are as follows.

```
imp1.dat
imp2.dat
imp3.dat
imp4.dat
imp5.dat
imp6.dat
imp7.dat
imp8.dat
imp9.dat
imp10.dat
imp11.dat
imp12.dat
imp13.dat
imp14.dat
imp15.dat
imp16.dat
imp17.dat
imp18.dat
imp19.dat
imp20.dat
```

The Mplus input file for analyzing the imputations is `Ex6.inp`. The script is virtually identical to the `Ex1.1.inp` file described in Section 1.2 with three exceptions. First, instead of naming the raw data set, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Finally, the `MODEL` section no longer specifies a normal distribution for the predictors. Readers can refer back to Section 1.2 for a detailed description of the other commands. The code block below shows the analysis and pooling script.

Mplus Script Ex6.inp

```
DATA:
file = ./imps/implist.dat;
type = imputation;
VARIABLE:
names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
  read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7;
usevariables = read9 read1 lrnprob1 behsymp1;
MODEL:
read9 on read1 lrnprob1 behsymp1 (beta1-beta3);
MODEL TEST:
0 = beta1; 0 = beta2; 0 = beta3;
OUTPUT:
stdyx cinterval;
```

6.8 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus F test to evaluate the set of slope coefficients. The `MODEL TEST` command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and p-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION
...

Wald Test of Parameter Constraints

          Value                            175.893

          Degrees of Freedom                     3
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
   READ1             0.506      0.043     11.868      0.000      0.182
   LRNPROB1         -0.231      0.113     -2.047      0.041      0.149
   BEHSYMP1         -0.189      0.101     -1.864      0.062      0.160

Intercepts
   READ9            65.487      5.803     11.284      0.000      0.150

Residual Variances
   READ9            86.366     11.202      7.710      0.000      0.138
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by .51 points on `READ9`. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 11.87$, $p < .001$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Specifying the `stdyx` keyword with the `OPTIONS` command prints the table of standardized estimates and *R*-squared statistics shown below. The slope coefficients convey the expected change in standard deviation units for a one standard deviation increase in each predictor. For example, the model predicts that two individuals who differ by one standard deviation on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by 0.70 standard deviations on `READ9`. Collectively, the predictors explain 61% of the variation in ninth-grade reading scores.

```
STANDARDIZED MODEL RESULTS

STDYX Standardization

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

 READ9    ON
    READ1            0.701      0.044     15.767      0.000      0.102
    LRNPROB1        -0.168      0.082     -2.036      0.042      0.157
    BEHSYMP1        -0.153      0.082     -1.861      0.063      0.159

 Intercepts
    READ9            4.424      0.531      8.332      0.000      0.152

 Residual Variances
    READ9            0.394      0.055      7.166      0.000      0.099

R-SQUARE

    Observed                                        Two-Tailed   Rate of
    Variable       Estimate       S.E.  Est./S.E.    P-Value    Missing

    READ9            0.606      0.055     11.033      0.000      0.099
```

6.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex6.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex6.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
```

```
3     /imputation_ id male hispanic riskgrp atrisk
4       behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
5       math1 math2 math3 math9 math9grp stanmath7.
6   EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 7 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 8 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 9.

SPSS Script Ex6.spwb, continued

```
7   SORT CASES by imputation_.
8   SPLIT FILE layered by imputation_.
9   REGRESSION
10    /descriptives mean stddev corr sig n
11    /dependent read9
12    /method enter read1 lrnprob1 behsymp1.
```

6.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (t = 11.73, p < .001). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	1	(Constant)	66.432	5.359		12.397	<.001			
		read1	.492	.038	.703	12.867	<.001			
		lrnprob1	-.283	.104	-.207	-2.712	.008			
		behsymp1	-.127	.092	-.105	-1.381	.170			
2.00	1	(Constant)	69.162	5.650		12.240	<.001			
		read1	.502	.042	.670	12.060	<.001			
		lrnprob1	-.296	.108	-.212	-2.737	.007			
		behsymp1	-.200	.097	-.160	-2.054	.042			
• • •										
20.00	1	(Constant)	66.690	5.270		12.654	<.001			
		read1	.493	.038	.701	12.823	<.001			
		lrnprob1	-.282	.104	-.211	-2.714	.008			
		behsymp1	-.134	.093	-.112	-1.442	.152			
Pooled	1	(Constant)	65.487	5.877		11.144	<.001	.146	.169	.993
		read1	.506	.043		11.725	<.001	.177	.212	.991
		lrnprob1	-.231	.114		-2.022	.043	.146	.168	.993
		behsymp1	-.189	.102		-1.841	.066	.156	.182	.992

a. Dependent Variable: read9

7

MCMC: Binary Logistic Regression

This example illustrates a binary logistic regression analysis with incomplete data. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READGRP_9$	9th grade reading classification	17.4	0 = Below average, 1 = Average
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

7.1 Analysis Model

The analysis model features a binary classification of ninth grade reading performance regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$\text{logit}(READGRP_9) = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) \quad (16)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Blimp automatically assigns a multivariate normal distribution to the predictors.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following the same factored regression specification from earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} READ_2 = \gamma_{01} + \gamma_{11}(READGRP_9) + \gamma_{21}(READ_1) + \ \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1) + \epsilon_1 \\ STANREAD_7 = \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READGRP_9) + \ \gamma_{32}(READ_1) \\ + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1) + \epsilon_2 \end{aligned} \quad (17)$$

Along with the logistic regression model from Equation 16, the collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram in Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

7.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex7.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex7.1.imp

```
1  DATA: behaviorachievement.dat;
2  VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
3     read1 read2 read3 read9 read9grp stanread7
4     math1 math2 math3 math9 math9grp stanmath7;
5  ORDINAL: read9grp;
6  MISSING: 999;
```

```
7   MODEL:
8   focal.model:
9   logit(read9grp) ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
10  auxiliary.models:
11  stanread7 read2 ~ read9grp read1 lrnprob1 behsymp1;
12  WALDTEST:beta1:beta3 = 0;
13  SEED: 90291;
14  BURN: 1000;
15  ITERATIONS: 10000;
```

The first six lines can be viewed as a set of commands that specify information about the data and variables. The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the `VARIABLES` command names the data columns. The `ORDINAL` command on line 5 defines the outcome as categorical. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The `MISSING` command on line 6 defines a global missing value code as 999.

The `MODEL` and `WALDTEST` blocks can be viewed as a set. The `MODEL` command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses labels (`focal.model` and `auxiliary.models`) to order output tables, such that the logistic model appears first followed by the auxiliary variable models. The focal model listed on line 9 assigns labels the slope coefficients using the @ symbol. Listing the dependent variable inside the `logit` function triggers logistic regression rather than the default probit regression. Blimp automatically configures the explanatory variable models under the assumption that they are normally distributed. Line 11 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 17; in the first model, `READ2` is regressed on the focal variables, and the second model features `STANREAD7` regressed on `READ2` and the focal variables. The `WALDTEST` command uses the parameter labels to specify a custom hypothesis test that all three slopes equal zero. This so-called Bayesian Wald test (Asparouhov & Muthén, 2021) is a frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test.

Finally, lines 13 through 15 can be viewed as a block of commands that specify features of the MCMC algorithm: the `SEED` command gives an integer string that initializes the random number generator, the `BURN` command specifies the number of iterations for the warm-up or burn-in period, and the `ITERATIONS` command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex7.R is shown below.

rblimp Script Ex7.R

```
library(rblimp)
load('behaviorachievement.rda')

mymodel <- rblimp(
  data = behaviorachievement,
  ordinal = 'read9grp ',
  model = '
    focal.model:
    logit(read9grp) ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
    auxiliary.models:
    stanread7 read2  ~ read9grp read1 lrnprob1 behsymp1',
  waldtest = 'beta1:beta3 = 0',
  seed = 90291,
  burn = 1000,
  iter = 10000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script. Finally, the output(mymodel) function prints the Blimp output.

7.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that

these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           26 to 50              1.140             2
                           51 to 100             1.072             2
                           76 to 150             1.041             3
                                 ...               ...           ...
                         451 to 900              1.009            37
                         476 to 950              1.008            19
                         501 to 1000             1.008            37
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3.

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test;

Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  logit(read9grp)
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Coefficients:							
Intercept	−2.721	1.291	−5.281	−0.204	4.452	0.035	4802.329
read1	0.062	0.013	0.038	0.090	22.534	0.000	2686.246
lrnprob1	−0.034	0.030	−0.095	0.024	1.303	0.254	3855.021
behsymp1	−0.021	0.026	−0.073	0.027	0.737	0.391	4501.123
Odds Ratio:							
Intercept	0.066	0.265	0.005	0.816	0.310	0.578	6757.096
read1	1.064	0.014	1.039	1.094	5744.907	0.000	2684.389
lrnprob1	0.966	0.029	0.909	1.025	1100.011	0.000	3857.606
behsymp1	0.979	0.025	0.929	1.027	1539.695	0.000	4502.785
Proportion Variance Explained							
by Coefficients	0.152	0.058	0.067	0.288	---	---	4354.701
by Residual Variation	0.848	0.058	0.712	0.933	---	---	4354.701

To begin, the `N_Eff` values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the `N_Eff` values are insufficient, increasing the value on the `ITERATIONS` command will remedy the issue. The table summarizing the focal regression model is shown below.

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` differ by 0.06. The 95% credible interval limits suggest this effect is statistically different from zero ($p <$

.05) because the null value is well outside the interval. The frequentist test statistic and *p*-value give the same conclusion. The printed output also includes the table of odds ratios that reflect multiplicative changes to the odds. For example, a one-point increase in first grade reading scores increases the odds of average or higher ninth grade reading by a factor 1.06, holding first grade learning problems and behavioral symptoms constant. Collectively, the predictors explain 15% of the variation in the underlying logistic latent variable. Note that the tabled values are numerically equivalent to the maximum likelihood estimates from Section 2.7.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

Finally, recall that the `WALDTEST` command requested a Bayesian Wald chi-square statistic (Asparouhov & Muthén, 2021) that evaluates the null hypothesis that all population slopes equal zero. To reiterate, the Wald test is frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test. The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT` section under the `WALD TEST` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT:

...

  WALD TESTS (Asparouhov & Muthén, 2021)

  Test #1

    Full:
      [1]  logit(read9grp) ~ Intercept read1@beta1 lrnprob1@beta2 behsymp1@beta3

    Restricted:
      [1]  logit(read9grp) ~ Intercept read1@beta1 lrnprob1@beta2 behsymp1@beta3
```

```
Constraints in Restricted:
  [1]  beta1 = 0
  [2]  beta2 = 0
  [3]  beta3 = 0


Wald Statistic (Chi-Square)                23.618
Number of Parameters Tested (df)                3
Probability                                 0.000
```

7.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex7.2.imp` is identical `Ex7.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they

appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
    read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex7.R

```
1   library(rblimp)
2   load('behaviorachievement.rda')
3
4   mymodel <- rblimp(
5     data = behaviorachievement,
6     ordinal = 'read9grp ',
7     model = '
8       focal.model:
9       logit(read9grp) ~ read1@beta1 lrnprob1@beta2 behsymp1@beta3;
10      auxiliary.models:
11      stanread7 read2  ~ read9grp read1 lrnprob1 behsymp1',
```

```
  waldtest = 'beta1:beta3 = 0',
  seed = 90291,
  burn = 1000,
  iter = 10000)
  nimps = 20,
  chains = 20)
output(mymodel)
```

The SAVE command is no longer necessary because imputations are automatically stored in a rblimp object called mymodel@imputations. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

7.5 Analyzing Multiple Imputations in R

Continuing with the previous rblimp script, the following excerpt from Ex7.R shows how to perform multiple imputation inference. The script requires the mitml package (Grund et al., 2023).

R Script Ex7.R

```
library(rblimp)
library(mitml)
load('behaviorachievement.rda')

mymodel <- rblimp(...)

implist <- as.mitml(mymodel)
fit <- with(implist,
  glm(read9grp ~ read1 + lrnprob1 + behsymp1, family = 'binomial')
estimates <- testEstimates(fit, extra.pars = T, df.com = 134)
estimates
confint(estimates)

null <- with(implist, glm(read9grp ~ 1, family = 'binomial'))
testModels(fit, null, df.com = 134, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 8 and 9 fits the focal regression model using the `lm` function, and line 10 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 11 and 12 print the pooled estimates and confidence intervals. Finally, lines 14 and 15 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 13 that represents the null hypothesis, which in this case is an empty regression model with just an intercept. On line 14, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

7.6 R Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    -3.602     1.672    -2.154    79.914     0.034     0.294     0.246
read1           0.068     0.015     4.463    93.795     0.000     0.206     0.188
lrnprob1       -0.029     0.030    -0.971    90.209     0.334     0.227     0.202
behsymp1       -0.019     0.025    -0.762   106.973     0.448     0.135     0.135

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.
```

```
                  2.5 %      97.5 %
(Intercept) -6.92951753 -0.27356800
read1        0.03751055  0.09763918
lrnprob1    -0.08980267  0.03082041
behsymp1    -0.06832451  0.03038596
```

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 differ by 0.07. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 4.46$, $p < .001$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation. Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets.
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
      7.359       3 120.633   0.000   0.214

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.
```

7.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). Section 6.7 shows the contents of this file.

The Mplus input file for analyzing the imputations is Ex7.inp. The script is similar to the Ex2.1.inp file described in Section 2.2 with three exceptions. First, instead of naming the raw data set, the DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand

instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Finally, the MODEL section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. Readers can refer to Section 2.2 for a detailed description of the other commands. The code block below shows the analysis and pooling script.

Mplus Script Ex7.inp

```
DATA:
file = ./imps/implist.dat;
type = imputation;
VARIABLE:
names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
  read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7;
usevariables = read9grp read1 lrnprob1 behsymp1;
categorical = read9grp;
ANALYSIS:
estimator = ml;
link = logit;
MODEL:
read9grp on read1 lrnprob1 behsymp1 (beta1-beta3);
MODEL TEST:
0 = beta1; 0 = beta2; 0 = beta3;
OUTPUT:
stdyx cinterval;
```

7.8 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus F test to evaluate the set of slope coefficients. The MODEL TEST command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and p-value appear near the bottom of the MODEL FIT INFORMATION section under the Wald Test of Parameter Constraints heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                        4

...

Wald Test of Parameter Constraints

          Value                            23.342
          Degrees of Freedom                    3
          P-Value                          0.0001
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS
                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9GRP   ON
    READ1            0.068      0.015      4.463      0.000      0.173
    LRNPROB1        -0.029      0.030     -0.971      0.331      0.188
    BEHSYMP1        -0.019      0.025     -0.762      0.446      0.121
 Thresholds
    READ9GRP$1       3.602      1.672      2.154      0.031      0.231
```

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` differ by 0.07. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 4.46$, $p < .001$). Note that Mplus reports a threshold parameter instead of the usual

regression intercept. The threshold from a binary logistic model has the same value but opposite sign as the intercept (i.e., $\hat{\beta}_0 = -3.60$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Finally, the printed output also includes the table of odds ratios that reflect multiplicative changes to the odds. For example, a one-point increase in first grade reading scores increases the odds of average or higher ninth grade reading by a factor 1.08, holding first grade learning problems and behavioral symptoms constant.

```
CONFIDENCE INTERVALS OF MODEL RESULTS

                 Lower 2.5%    Lower 5%    Estimate    Upper 5%  Upper 2.5%%

READ9GRP ON
    READ1             0.038       0.043       0.068       0.092       0.097
    LRNPROB1         -0.089      -0.079      -0.029       0.020       0.030
    BEHSYMP1         -0.068      -0.060      -0.019       0.022       0.030

 Thresholds
    READ9GRP$1        0.324       0.851       3.602       6.352       6.879

CONFIDENCE INTERVALS FOR THE LOGISTIC REGRESSION ODDS RATIO RESULTS

 READ9GRP ON
    READ1             1.039       1.044       1.070       1.097       1.102
    LRNPROB1          0.915       0.924       0.971       1.021       1.030
    BEHSYMP1          0.934       0.942       0.981       1.022       1.030
```

7.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex7.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex7.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3    /imputation_ id male hispanic riskgrp atrisk
4     behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
5     math1 math2 math3 math9 math9grp stanmath7.
6  EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 7 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 10 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 9.

SPSS Script Ex7.spwb, continued

```
7   SORT CASES by imputation_.
8   SPLIT FILE layed by imputation_.
9   LOGISTIC REGRESSION read9grpNew
10    /method=enter read1 lrnprob1 behsymp1.
```

7.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data logistic regression analysis. For example, consider the first-grade reading score slope. The model predicts that the logits for two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 differ by 0.07. SPSS does not report the corresponding test statistic for the slope coefficient, but it does include the p-value for the test statistic which indicates that the slope coefficient is statistically different from zero ($p < .001$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Variables in the Equation

imputation_			B	S.E.	Wald	df	Sig.	Exp(B)	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	Step 1[a]	read1	.060	.013	22.145	1	.000	1.061			
		lrnprob1	-.021	.026	.658	1	.417	.979			
		behsymp1	-.017	.023	.557	1	.456	.983			
		Constant	-3.385	1.411	5.753	1	.016	.034			
2.00	Step 1[a]	read1	.070	.014	25.125	1	.000	1.073			
		lrnprob1	-.025	.028	.826	1	.363	.975			
		behsymp1	-.020	.023	.746	1	.388	.980			
		Constant	-3.967	1.499	7.006	1	.008	.019			
• • •											
20.00	Step 1[a]	read1	.060	.013	21.439	1	.000	1.061			
		lrnprob1	-.047	.028	2.841	1	.092	.954			
		behsymp1	-.011	.023	.248	1	.619	.989			
		Constant	-2.496	1.416	3.107	1	.078	.082			
Pooled	Step 1[a]	read1	.068	.015			.000	1.070	.173	.206	.991
		lrnprob1	-.029	.030			.332	.971	.188	.227	.991
		behsymp1	-.019	.025			.446	.981	.121	.135	.994
		Constant	-3.602	1.672			.032	.027	.231	.294	.989

a. Variable(s) entered on step 1: read1, lrnprob1, behsymp1.

8

MCMC: Regression With Binary and Ordinal Predictors

This example illustrates a multiple regression analysis with incomplete categorical predictors. The analysis uses the `mathachievement.dat` data set taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
MATHPOST	Math achievement posttest	18.0	Numeric
CONDITION	Experimental condition	0	0 = Comparison, 1 = Intervention
FRLUNCH	Lunch assistance code	4.4	0 = None, 1 = Free/reduced lunch
EFFICACY	Math self-efficacy rating	9.6	Ordinal (1 to 6)
MATHPRE	Math achievement pretest	0	Numeric
	Auxiliary Variables		
ATRISK	Behavioral disorder risk	5.2	0 = Low risk, 1 = At-risk
STANREAD	Standardized reading	9.2	Numeric

8.1 Analysis Model

The analysis model features math posttest scores regressed on the experimental condition and lunch assistance dummy codes, math self-efficacy ratings, and math pretest scores.

$$MATHPOST = \beta_0 + \beta_1(CONDITION) + \beta_2(FRLUNCH) + \beta_3(EFFICACY) + \beta_4(MATHPRE) + \varepsilon \quad (18)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. In this case, the predictor set includes incomplete binary and ordinal variables. Blimp uses a probit regression formulation that envisions discrete responses as arising from underlying continuous latent response variables. The software assumes that continuous predictors and the latent response variables are multivariate normal.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} ATRISK^* &= \gamma_{03} + \gamma_{13}(MATHPOST) + \gamma_{23}(CONDITION) \\ &+ \gamma_{33}(FRLUNCH) + \gamma_{43}(EFFICACY) + \gamma_{53}(MATHPRE) + \epsilon_3 \\ STANREAD &= \gamma_{04} + \gamma_{14}(ATRISK) + \gamma_{24}(MATHPOST) + \gamma_{34}(CONDITION) \\ &+ \gamma_{44}(FRLUNCH) + \gamma_{54}(EFFICACY) + \gamma_{64}(MATHPRE) + \epsilon_4 \end{aligned} \quad (19)$$

The `ATRISK` model is a probit regression, with the binary outcome model as a latent response variable (denoted by the asterisk superscript). Again, the entire collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

8.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex8.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex8.1.imp

```
1  DATA: mathachievement.dat;
2  VARIABLES: id condition male frlunch atrisk stanread efficacy anxiety
```

```
3      mathpre mathpost;
4   ORDINAL: condition frlunch atrisk efficacy;
5   MISSING: 999;
6   FIXED: condition mathpre;
7   MODEL:
8   focal.model:
9   mathpost ~ condition@beta1 frlunch@beta2 efficacy@beta3 mathpre@beta4;
10  auxiliary.models:
11  stanread atrisk ~ mathpost condition frlunch efficacy mathpre;
12  WALDTEST: beta1:beta4 = 0;
13  SEED: 90291;
14  BURN: 5000;
15  ITERATIONS: 10000;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the `VARIABLES` command names the data columns. The `ORDINAL` command on line 4 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The `MISSING` command on line 5 defines a global missing value code as 999.

The `FIXED`, `MODEL`, and `WALDTEST` blocks can be viewed as a set. The `FIXED` command identifies the two complete variables, which do not require a distribution or regression model. Beginning on line 7, the `MODEL` command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses labels (`focal.model` and `auxiliary.models`) to order output tables, such that the focal model appears first followed by the auxiliary variable models. The focal model listed on line 9 assigns labels the slope coefficients using the @ symbol. Blimp automatically configures the explanatory variable models under the assumption that they are normally distributed. Line 11 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 19; in the first model, `READ2` is regressed on the focal variables, and the second model features `STANREAD7` regressed on `READ2` and the focal variables. The `WALDTEST` command uses the parameter labels to specify a custom hypothesis test that all three slopes equal zero. This so-called Bayesian Wald test (Asparouhov & Muthén, 2021) is a frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test.

Finally, lines 13 through 15 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex8.R is shown below.

rblimp Script Ex8.R

```
1   library(rblimp)
2   load(mathachievement.rda')
3
4   mymodel <- rblimp(
5     data = mathachievement,
6     ordinal = 'condition frlunch atrisk efficacy',
7     fixed = 'condition mathpre',
8     model = '
9       focal.model:
10      mathpost ~ condition@beta1 frlunch@beta2 efficacy@beta3 mathpre@beta4;
11      auxiliary.models:
12      stanread atrisk ~ mathpost condition frlunch efficacy mathpre',
13    waldtest = 'beta1:beta4 = 0',
14    seed = 90291,
15    burn = 5000,
16    iter = 10000)
17  output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script. Finally, the output(mymodel) function prints the Blimp output.

8.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          126 to 250             1.416            59
                          251 to 500             1.425            57
                          376 to 750             1.146            57
                                 ...               ...            ..
                         2251 to 4500            1.040            57
                         2376 to 4750            1.016            56
                         2501 to 5000            1.009            56
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3.

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior

medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  mathpost

Parameters                    Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
Variances:
  Residual Var.               53.303      5.509     43.687     65.369        ---        ---   5636.603

Coefficients:
  Intercept                   28.345      3.088     22.308     34.514     84.418      0.000   7012.232
  condition                    2.263      1.047      0.202      4.311      4.644      0.031   6953.241
  frlunch                     -5.502      1.095     -7.608     -3.325     25.087      0.000   5344.564
  efficacy                     0.831      0.346      0.159      1.517      5.773      0.016   4712.974
  mathpre                      0.530      0.062      0.408      0.653     71.920      0.000   6537.163

Standardized Coefficients:
  condition                    0.117      0.054      0.010      0.222      4.715      0.030   6897.665
  frlunch                     -0.281      0.052     -0.378     -0.173     28.538      0.000   5652.427
  efficacy                     0.139      0.057      0.027      0.248      5.904      0.015   4706.081
  mathpre                      0.477      0.048      0.378      0.564     98.308      0.000   6400.853

Proportion Variance Explained
  by Coefficients              0.426      0.046      0.328      0.510        ---        ---   5954.279
  by Residual Variation        0.574      0.046      0.490      0.672        ---        ---   5954.279

-------------------------------------------------------------------------------------------------------
```

To begin, the `N_Eff` values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent

estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue.

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, consider the slope for the treatment assignment dummy code. The positive coefficient indicates that, for two students who share the same covariate profile (i.e., lunch assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.26 points higher than the student in the control group. The 95% credible interval limits suggest this effect is statistically different from zero ($p <$ .05) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. Note that the tabled values are numerically equivalent to the maximum likelihood estimates from Section 3.3.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in OUTCOME MODEL ESTIMATES section with the focal results, and the auto-generated predictor models are displayed under the heading PREDICTOR MODEL ESTIMATES. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

Finally, recall that the WALDTEST command requested a Bayesian Wald chi-square statistic (Asparouhov & Muthén, 2021) that evaluates the null hypothesis that all population slopes equal zero. To reiterate, the Wald test is frequentist chi-square statistic that mimics its likelihood-based counterpart, but MCMC generates the point estimates and "standard errors" for the test. The chi-square statistic, degrees of freedom, and p-value appear near the bottom of the MODEL FIT section under the WALD TEST heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT:

  ...

  WALD TESTS (Asparouhov & Muthén, 2021)
```

```
Test #1

  Full:
    [1]  mathpost ~ Intercept condition@beta1 frlunch@beta2 efficacy@beta3
                mathpre@beta4

  Restricted:
    [1]  mathpost ~ Intercept condition@beta1 frlunch@beta2 efficacy@beta3
                mathpre@beta4

  Constraints in Restricted:
    [1]  beta1 = 0
    [2]  beta2 = 0
    [3]  beta3 = 0
    [4]  beta4 = 0


  Wald Statistic (Chi-Square)              142.310
  Number of Parameters Tested (df)               4
  Probability                                0.000
```

8.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex8.2.imp` is identical `Ex8.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding

autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     id condition male frlunch atrisk stanread efficacy anxiety
     mathpre mathpost

   stacked = './imps/imps.dat'

     imp# id condition male frlunch atrisk stanread efficacy
     anxiety mathpre mathpost
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In rblimp, the NIMPS and CHAINS commands are added as input parameters to the function as follows.

rblimp Script Ex8.R

```
library(rblimp)
load('mathachievement.rda')

mymodel <- rblimp(
  data = mathachievement,
  ordinal = 'condition frlunch atrisk efficacy',
  fixed = 'condition mathpre',
  model = '
    focal.model:
    mathpost ~ condition@beta1 frlunch@beta2 efficacy@beta3 mathpre@beta4;
    auxiliary.models:
    stanread atrisk ~ mathpost condition frlunch efficacy mathpre',
  waldtest = 'beta1:beta4 = 0',
  seed = 90291,
  burn = 5000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

8.5 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex8.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex8.R

```
library(rblimp)
library(mitml)
```

```
3   load('mathachievement.rda')
4
5   mymodel <- rblimp(...)
6
7   implist <- as.mitml(mymodel)
8   fit <- with(implist,
9      lm(mathpost ~ condition + frlunch + efficacy + mathpre))
10  estimates <- testEstimates(fit, extra.pars = T, df.com = 245)
11  estimates
12  confint(estimates)
13
14  null <- with(implist, lm(mathpost ~ 1))
15  testModels(fit, null, df.com = 245, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 8 and 9 fit the focal regression model using the `lm` function, and line 10 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 11 and 12 print the pooled estimates and confidence intervals. Finally, lines 14 and 15 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 13 that represents the null hypothesis, which in this case is an empty regression model with just an intercept. On line 14, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

8.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-

data sampling variation, and the FMI column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    28.302     3.183     8.892   125.465     0.000     0.290     0.237
condition       2.206     1.055     2.091   133.634     0.038     0.263     0.220
frlunch        -5.392     1.094    -4.928   110.541     0.000     0.348     0.271
efficacy        0.832     0.356     2.339   103.454     0.021     0.380     0.289
mathpre         0.532     0.063     8.448   128.385     0.000     0.280     0.231

                  Estimate
Residual~~Residual   53.133

Hypothesis test adjusted for small samples with df=[245]
complete-data degrees of freedom.

                 2.5 %     97.5 %
(Intercept) 22.0026367 34.6005103
condition    0.1192603  4.2925545
frlunch     -7.5596123 -3.2237303
efficacy     0.1266288  1.5376826
mathpre      0.4071878  0.6562495
```

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, consider the slope for the treatment assignment dummy code. The positive coefficient indicates that, for two students who share the same covariate profile (i.e., lunch assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.21 points higher than the student in the control group. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (t = 2.09, p = .04). Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation. Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets.
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     33.796       4 197.183   0.000   0.332

Hypothesis test adjusted for small samples with df=[245]
complete-data degrees of freedom.
```

8.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). Section 6.7 shows the contents of this file.

The Mplus input file for analyzing the imputations is `Ex8.inp`. The script is like the `Ex3.inp` file described in Section 3.2 with three exceptions. First, instead of naming the raw data set, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Finally, the `MODEL` section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. Readers can refer back to Section 3.2 for a detailed description of the other commands. The code block below shows the analysis and pooling script.

Mplus Script Ex8.inp

```
1  DATA:
2  file = ./imps/implist.dat;
3  type = imputation;
4  VARIABLE:
5  names = id condition male frlunch atrisk stanread
6    efficacy anxiety mathpre mathpost;
7  usevariables = mathpost condition frlunch efficacy mathpre;
8  MODEL:
```

```
 9  mathpost on condition frlunch efficacy mathpre (beta1-beta4);
10  MODEL TEST:
11  0 = beta1; 0 = beta2; 0 = beta3; 0 = beta4;
12  OUTPUT:
13  stdyx cinterval;
```

8.8 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus *F* test to evaluate the set of slope coefficients. The `MODEL TEST` command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                        6

...

Wald Test of Parameter Constraints

          Value                            125.646
          Degrees of Freedom                     4
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS
```

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value	Rate of Missing
MATHPOST ON					
CONDITION	2.206	1.047	2.107	0.035	0.215
FRLUNCH	-5.392	1.086	-4.965	0.000	0.267
EFFICACY	0.832	0.353	2.356	0.018	0.285
MATHPRE	0.532	0.062	8.515	0.000	0.226
Intercepts					
MATHPOST	28.301	3.158	8.962	0.000	0.233
Residual Variances					
MATHPOST	52.070	5.500	9.467	0.000	0.287

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, consider the slope for the treatment assignment dummy code. The positive coefficient indicates that, for two students who share the same covariate profile (i.e., lunch assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.21 points higher than the student in the control group. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 2.11$, $p = .04$). Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation. The output also includes a table with standardized coefficients and the *R*-squared statistic.

8.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex8.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex8.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3   /imputation_ id condition male frlunch atrisk stanread efficacy anxiety
4    mathpre mathpost.
5  EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 6 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 7 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 8.

SPSS Script Ex8.spwb, continued

```
6   SORT CASES by imputation_.
7   SPLIT FILE layered by imputation_.
8   REGRESSION
9     /descriptives mean stddev corr sig n
10    /dependent mathpost
11    /method enter condition frlunch efficacy mathpre.
```

8.8 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in

variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data regression analysis with categorical predictors. For example, consider the slope for the treatment assignment dummy code. The positive coefficient indicates that, for two students who share the same covariate profile (i.e., lunch assistance, self-efficacy, and pretest scores), the model predicts that the student in the experimental condition should score 2.21 points higher than the student in the control group. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (t = 2.09, p = .04). Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients		Standardized Coefficients			Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
			B	Std. Error	Beta	t	Sig.			
1.00	1	(Constant)	26.766	2.892		9.256	.000			
		condition	2.677	.971	.134	2.756	.006			
		frlunch	-6.281	.977	-.309	-6.429	.000			
		efficacy	.566	.310	.092	1.823	.069			
		mathpre	.575	.058	.498	9.864	.000			
2.00	1	(Constant)	28.064	2.819		9.955	.000			
		condition	2.551	.943	.132	2.704	.007			
		frlunch	-5.846	.950	-.298	-6.153	.000			
		efficacy	.637	.311	.104	2.046	.042			
		mathpre	.549	.056	.492	9.784	.000			
• • •										
20.00	1	(Constant)	28.652	2.774		10.328	.000			
		condition	2.067	.929	.111	2.224	.027			
		frlunch	-4.662	.931	-.247	-5.006	.000			
		efficacy	.821	.307	.138	2.672	.008			
		mathpre	.526	.055	.489	9.528	.000			
Pooled	1	(Constant)	28.302	3.183		8.892	.000	.229	.290	.989
		condition	2.206	1.055		2.091	.037	.212	.263	.990
		frlunch	-5.392	1.094		-4.928	.000	.263	.348	.987
		efficacy	.832	.356		2.339	.020	.281	.380	.986
		mathpre	.532	.063		8.448	.000	.223	.280	.989

a. Dependent Variable: mathpost

9

MCMC: Regression With Multicategorical Predictors

This example illustrates a multiple regression analysis with an incomplete multicategorical predictor. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
RISKGRP	Emotional/behavioral disorder risk	2.2	1 = Low, 2 = Medium, 3 = High
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

9.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement, teacher-rated learning problems and behavioral symptoms, and a three-category nominal variable indicating risk for emotional or behavioral disorders.

$$\begin{aligned} READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1^*) \\ + \beta_4(MEDRISK) + \beta_5(HIGHRISK) + \varepsilon \end{aligned} \tag{20}$$

The `MEDRISK` and `HIGHRISK` variables are dummy code variables that contrast the medium- and high-risk groups, respectively, against the low-risk reference group. Blimp uses a probit regression formulation that envisions multicategorical variables as arising from underlying continuous latent response difference scores. The software automatically assumes that continuous predictors and the latent response variables are multivariate normal.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following the same factored regression specification from earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} READ_2 = \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1^*) \\ + \gamma_{51}(MEDRISK) + \gamma_{61}(HIGHRISK) + \epsilon_1 \\ STANREAD_7 = \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \gamma_{32}(READ_1) \\ + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1^*) + \gamma_{62}(MEDRISK) + \gamma_{72}(HIGHRISK) + \epsilon_2 \end{aligned} \tag{21}$$

Along with the focal regression model from Equation 20, the collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

9.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex9.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex9.1.imp

```
1  DATA: behaviorachievement.dat;
2  VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
3    read1 read2 read3 read9 read9grp stanread7 math1 math2
4    math3 math9 math9grp stanmath7;
5  NOMINAL: riskgrp;
6  MISSING: 999;
7  MODEL:
8  focal.model:
9  read9 ~ read1 lrnprob1 behsymp1 riskgrp;
10 auxiliary.models:
11 stanread7 read2 ~ read9 read1 lrnprob1 behsymp1 riskgrp;
12 SEED: 90291;
13 BURN: 2000;
14 ITERATIONS: 10000;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The NOMINAL command on line 5 identifies the multicategorical nominal predictor. By default, the group with the lowest numeric code serves as the reference category (in this example, 1 = low risk), and the user can change this specification if desired. The MISSING command on line 6 defines a global missing value code as 999.

The MODEL and WALDTEST blocks can be viewed as a set. Beginning on line 7, the MODEL command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses labels (focal.model and auxiliary.models) to order output tables, such that the focal model appears first followed by the auxiliary variable models. The focal model listed on line 9 includes the multicategorical nominal variable, which Blimp represents as a pair of dummy codes. Blimp automatically configures the explanatory variable models under the assumption that the numeric predictors and latent response variables are normally distributed. Line 11 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 21; in the first model, READ2 is regressed on the focal variables, and the second model features STANREAD7 regressed on READ2 and the focal variables.

Finally, lines 12 through 14 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

Previous examples assigned labels to slope coefficients using the @ symbol, and these labels were subsequently used in the WALDTEST command to specify custom hypothesis tests. With a multicategorical nominal predictor, it is necessary to attach labels to individual dummy codes. To do this, you list the nominal variable's name followed by a period and a numeric suffix with each category's code value. For example, line 9 in the script would be modified as follows

```
9    read9 ~ read1@b1 lrnprob1@b2 behsymp1@b3 riskgrp.2@b4 riskgrp.3@b5;
```

where RISKGRP.2 and RISKGRP.3 reference the two dummy variables for the groups coded 2 and 3 in the data. The WALDTEST command would then be constructed following earlier examples.

The corresponding rblimp script Ex9.R is shown below.

rblimp Script Ex9.R

```
1   library(rblimp)
2   load('behaviorachievement.rda')
3
4   mymodel <- rblimp(
5     data = behaviorachievement,
6     nominal = 'riskgrp',
7     model = '
8       focal.model:
9       read9 ~ read1 lrnprob1 behsymp1 riskgrp;
10      auxiliary.models:
11      stanread7 read2   ~ read9 read1 lrnprob1 behsymp1 riskgrp',
12    seed = 90291,
13    burn = 2000,
14    iter = 10000)
15  output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp` function. The two exceptions are the `VARIABLES` and `MISSING` commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the `MODEL` command) are separated by semicolons, as they are in the Blimp script. Finally, the `output(mymodel)` function prints the Blimp output.

9.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           51 to 100             1.305            77
                          101 to 200             1.200            62
                          151 to 300             1.064            56
                                 ...               ...            ..
                          901 to 1800            1.017            53
                          951 to 1900            1.014            53
                         1001 to 2000            1.017            56
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3.

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  read9

Parameters                      Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
Variances:
  Residual Var.                 91.703     13.152     70.459    121.914        ---        ---   5589.273

Coefficients:
  Intercept                     68.621      6.614     55.479     81.676    107.763      0.000   5703.283
  read1                          0.484      0.049      0.389      0.581     98.268      0.000   7086.883
  lrnprob1                      -0.250      0.121     -0.485     -0.007      4.205      0.040   5583.683
  behsymp1                      -0.170      0.107     -0.379      0.041      2.528      0.112   6010.276
  riskgrp.2                     -1.682      1.991     -5.632      2.220      0.707      0.401   7073.237
  riskgrp.3                     -2.814      2.707     -8.233      2.511      1.084      0.298   6138.228

Standardized Coefficients:
  read1                          0.658      0.052      0.544      0.751    155.706      0.000   6469.725
```

```
   lrnprob1                      -0.178    0.085    -0.340    -0.005     4.277     0.039  5544.858
   behsymp1                      -0.137    0.085    -0.300     0.033     2.566     0.109  5901.722
   riskgrp.2                     -0.055    0.065    -0.182     0.073     0.714     0.398  7086.836
   riskgrp.3                     -0.079    0.075    -0.225     0.072     1.094     0.296  6182.638

Proportion Variance Explained
   by Coefficients                0.599    0.050     0.488     0.684       ---       ---  5849.961
   by Residual Variation          0.401    0.050     0.316     0.512       ---       ---  5849.961

-------------------------------------------------------------------------------
```

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. The two dummy codes appear as RISKGRP.2 and RISKGRP.3, where the numeric suffices correspond to the numeric codes from the data. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 2.81 points lower than a low-risk student in the comparison group.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in OUTCOME MODEL ESTIMATES section with the focal results, and the auto-generated predictor models are displayed under the heading PREDICTOR MODEL ESTIMATES. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

9.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A

subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex9.2.imp` is identical `Ex9.1.imp`, but it adds the following lines.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
```

```
read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex9.R

```
library(rblimp)
load('behaviorachievement.rda')

mymodel <- rblimp(
  data = behaviorachievement,
  nominal = 'riskgrp',
  model = '
    focal.model:
    read9 ~ read1 lrnprob1 behsymp1 riskgrp;
    auxiliary.models:
    stanread7 read2   ~ read9 read1 lrnprob1 behsymp1 riskgrp',
  seed = 90291,
  burn = 2000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

9.5 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex9.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex9.R

```
1   library(rblimp)
2   library(mitml)
3   load('behaviorachievement.rda')
4
5   mymodel <- rblimp(...)
6
7   implist <- as.mitml(mymodel)
8   fit <- with(implist,
9     lm(read9 ~ read1 + lrnprob1 + behsymp1 + factor(riskgrp)))
10  estimates <- testEstimates(fit, extra.pars = T, df.com = 132)
11  estimates
12  confint(estimates)
13  null <- with(implist, lm(read9 ~ 1))
14  testModels(fit, null, df.com = 132, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 8 and 9 fit the focal regression model using the `lm` function, and line 10 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 11 and 12 print the pooled estimates and confidence intervals. Finally, lines 14 and 15 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 13 that represents the null hypothesis, which in this case is an empty regression model with just an intercept. On line 14, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have

resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

9.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    69.174     6.337    10.916    98.577     0.000     0.172     0.164
read1           0.477     0.048     9.928   103.392     0.000     0.146     0.144
lrnprob1       -0.250     0.117    -2.133    94.276     0.036     0.196     0.181
behsymp1       -0.166     0.108    -1.539    81.473     0.128     0.276     0.235
riskgrp2       -1.710     1.921    -0.890   116.647     0.375     0.079     0.088
riskgrp3       -3.115     2.867    -1.087    72.059     0.281     0.348     0.278

                   Estimate
Residual~~Residual   89.403

Hypothesis test adjusted for small samples with df=[132]
complete-data degrees of freedom.

                 2.5 %      97.5 %
(Intercept) 56.5999199 81.74779632
read1        0.3820806  0.57283035
lrnprob1    -0.4822008 -0.01730107
behsymp1    -0.3796984  0.04849960
riskgrp2    -5.5147538  2.09562507
riskgrp3    -8.8300720  2.59967016
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals

who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 9.93$, $p < .001$). The two dummy codes appear as RISKGRP2 and RISKGRP3. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 3.12 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation. Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets.
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     33.252       5 123.203   0.000   0.213

Hypothesis test adjusted for small samples with df=[132]
complete-data degrees of freedom.
```

9.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). Section 6.7 shows the contents of this file.

The Mplus input file for analyzing the imputations is Ex9.inp. The script is like previous Mplus scripts (e.g., the Ex1.1.inp file described in Section 1.2) with four exceptions. First, instead of naming the raw data set, the DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Third, the MODEL section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. Finally, lines 9 through 13 use the DEFINE command to create a pair of

dummy codes. Lines 10 and 11 initialize a pair of new variables (RISKGRP2 and RISKGRP3) with all 0s, and lines 12 and 13 recode these variables into dummy variables. Importantly, new variables computed with the DEFINE command must appear at the end of the usevariables list on line 8. The code block below shows the analysis and pooling script.

Mplus Script Ex9.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
6     read1 read2 read3 read9 read9grp stanread7
7     math1 math2 math3 math9 math9grp stanmath7;
8   usevariables = read9 read1 lrnprob1 behsymp1 riskgrp2 riskgrp3;
9   DEFINE:
10  riskgrp2 = 0;
11  riskgrp3 = 0;
12  if(riskgrp eq 2) then riskgrp2 = 1;
13  if(riskgrp eq 3) then riskgrp3 = 1;
14  MODEL:
15  read9 on read1 lrnprob1 behsymp1 riskgrp2 riskgrp3 (beta1-beta5);
16  MODEL TEST:
17  0 = beta1; 0 = beta2; 0 = beta3;
18  OUTPUT:
19  stdyx cinterval;
```

9.8 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus *F* test to evaluate the set of slope coefficients. The MODEL TEST command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the MODEL FIT INFORMATION section under the Wald Test of Parameter Constraints heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       7

...

Wald Test of Parameter Constraints

          Value                            173.432
          Degrees of Freedom                     5
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
   READ1             0.477      0.047     10.122      0.000      0.134
   LRNPROB1         -0.250      0.115     -2.173      0.030      0.172
   BEHSYMP1         -0.166      0.106     -1.566      0.117      0.228
   RISKGRP2         -1.710      1.882     -0.908      0.364      0.076
   RISKGRP3         -3.115      2.820     -1.105      0.269      0.272

Intercepts
   READ9            69.174      6.218     11.125      0.000      0.154

Residual Variances
   READ9            85.516     11.867      7.206      0.000      0.249
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (z = 10.29, p < .001). The two dummy codes appear as RISKGRP2 and RISKGRP3. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 3.12 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC estimation. The output also includes a table with standardized coefficients and the R-squared statistic.

9.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the stacked keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is Ex9.spwb. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex9.spwb

```
CD '/users/username/desktop'.
DATA LIST free file =  'imps.dat'
 /imputation_ id male hispanic riskgrp atrisk
  behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7.
EXE.

COMPUTE riskgrp2 = 0.
COMPUTE riskgrp3 = 0.
IF (riskgrp = 2) riskgrp2 = 1.
IF (riskgrp = 3) riskgrp3 = 1.
EXE.
```

The first line uses the CD command to change the working directory to the desktop. The username portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. The dummy codes for the RISKGRP variable are created beginning at line 8.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 13 sorts the data by the imputation index variable, and the SPLIT FILE command on line 14 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 15.

SPSS Script Ex9.spwb, continued

```
13  SORT CASES by imputation_.
14  SPLIT FILE layered by imputation_.
15  REGRESSION
16    /descriptives mean stddev corr sig n
17    /dependent read9
18    /method enter read1 lrnprob1 behsymp1 riskgrp2 riskgrp3.
```

9.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically

different from zero ($t = 9.93$, $p < .001$). The two dummy codes appear as `RISKGRP2` and `RISKGRP3`. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 3.12 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	1	(Constant)	69.665	5.345		13.033	.000			
		read1	.481	.041	.685	11.713	.000			
		lrnprob1	-.226	.099	-.165	-2.283	.024			
		behsymp1	-.214	.087	-.178	-2.453	.015			
		riskgrp2	-2.057	1.703	-.070	-1.208	.229			
		riskgrp3	-3.247	2.308	-.093	-1.407	.162			
2.00	1	(Constant)	65.652	6.102		10.760	.000			
		read1	.492	.047	.661	10.372	.000			
		lrnprob1	-.273	.111	-.197	-2.452	.016			
		behsymp1	-.096	.097	-.080	-.987	.325			
		riskgrp2	-1.477	1.907	-.050	-.774	.440			
		riskgrp3	-3.100	2.561	-.088	-1.210	.228			
• • •										
20.00	1	(Constant)	71.467	6.073		11.768	.000			
		read1	.477	.047	.659	10.087	.000			
		lrnprob1	-.249	.114	-.178	-2.186	.031			
		behsymp1	-.226	.100	-.184	-2.264	.025			
		riskgrp2	-1.486	1.940	-.049	-.766	.445			
		riskgrp3	-1.177	2.609	-.033	-.451	.653			
Pooled	1	(Constant)	69.174	6.337		10.916	.000	.149	.172	.993
		read1	.477	.048		9.928	.000	.129	.146	.994
		lrnprob1	-.250	.117		-2.133	.033	.166	.196	.992
		behsymp1	-.166	.108		-1.539	.125	.220	.276	.989
		riskgrp2	-1.710	1.921		-.890	.374	.073	.079	.996
		riskgrp3	-3.115	2.867		-1.087	.278	.263	.348	.987

a. Dependent Variable: read9

10

MCMC: Moderated Regression With an Interaction

This example illustrates a multiple regression analysis with an incomplete interaction effect. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing	Scale
	Focal Analysis Variables		
ATRISK	Emotion/behavior disorder risk	2.2%	0 = Low risk, 1 = At risk
$LRNPROB_1$	1st grade learning problems	2.2%	Numeric (31 to 88)
$READ_1$	1st grade broad reading	6.5%	Numeric (39 to 153)
$READ_9$	9th grade broad reading	17.4%	Numeric (41 to 123)
	Auxiliary Variables		
$READ_2$	2nd grade broad reading	9.4%	Numeric (20 to 150)
$STANREAD_7$	7th grade standardized reading	19.6%	Numeric (100 to 399)

10.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement, teacher-rated learning problems, and the product of first grade reading scores and learning problems, and a binary risk indicator.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(READ_1)(LRNPROB_1) + \beta_4(ATRISK) + \varepsilon \quad (22)$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Moderated regression models (and models with non-linearities more generally) require a factored regression specification that assigns separate distributions to the predictors and outcome. By default, Blimp invokes a multivariate normal distribution for incomplete predictors. Importantly, the product term does not require a unique distribution, as missing data imputation generates lower-order variables that preserve the interaction effect in the focal model.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$READ_2 = \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \gamma_{31}(LRNPROB_1) + \gamma_{41}(ATRISK) + \epsilon_4$$
$$STANREAD_7 = \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \gamma_{32}(READ_1) + \gamma_{42}(LRNPROB_1) + \gamma_{52}(ATRISK) + \epsilon_5 \quad (23)$$

Along with the other models, the collection of regression equations can be viewed as a path model where the focal analysis is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

10.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex10.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex10.1.imp

```
1  DATA: behaviorachievement.dat;
2  VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
3    read1 read2 read3 read9 read9grp stanread7
4    math1 math2 math3 math9 math9grp stanmath7;
5  ORDINAL: atrisk;
6  MISSING: 999;
7  CENTER: read1 lrnprob1 atrisk;
8  MODEL:
9  focal.model:
10 read9 ~ read1 lrnprob1 read1*lrnprob1 atrisk;
11 auxiliary.model:
12 stanread7 read2 ~ read9 read1 lrnprob1 atrisk;
13 SIMPLE: read1 | lrnprob1;
14 SEED: 90291;
15 BURN: 5000;
16 ITERATIONS: 10000;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the `VARIABLES` command names the data columns, the `ORDINAL` command identifies the binary risk indicator, and `MISSING` command on line 6 defines a global missing value code as 999.

The `CENTER`, `MODEL`, and `SIMPLE` blocks can be viewed as a set. The `CENTER` command deviates the two interacting variables at their iteratively-estimated grand means. Beginning on line 8, the `MODEL` command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses labels (`focal.model` and `auxiliary.models`) to order output tables, such that the focal model appears first followed by the auxiliary variable models. The focal model listed on line 10 includes a product term, which is specified by joining two variables with an asterisk. Blimp automatically configures the explanatory variable models under the assumption that they are normally distributed. Line 12 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 23; in the first model, `READ2` is regressed on the focal variables, and the second model features `STANREAD7` regressed on `READ2` and the focal variables. The `SIMPLE` command requests the conditional effects (i.e., simple slopes) of `READ1` at different levels of `LRNPROB1`. By default, Blimp adopts a pick-a-point approach that uses standard deviation units of the moderator variable, although the user can specify custom values. Finally,

lines 14 through 16 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex10.R is shown below.

rblimp Script Ex10.R

```
1  library(rblimp)
2  load('behaviorachievement.rda')
3
4  mymodel <- rblimp(
5    data = behaviorachievement,
6    ordinal = 'atrisk',
7    center = 'read1 lrnprob1 atrisk',
8    model = '
9      focal.model:
10     read9 ~ read1 lrnprob1 read1*lrnprob1 atrisk ;
11     auxiliary.models:
12     stanread7 read2  ~ read9 read1 lrnprob1 atrisk',
13   simple = 'read1 | lrnprob1',
14   seed = 90291,
15   burn = 5000,
16   iter = 10000)
17 output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script. Finally, the output(mymodel) function prints the Blimp output.

10.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          126 to 250             1.116            39
                          251 to 500             1.127            48
                          376 to 750             1.033            39
                                 ...               ...            ..
                         2251 to 4500            1.015            48
                         2376 to 4750            1.021            48
                         2501 to 5000            1.014            48
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3.

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior

medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  read9

Grand Mean Centered: atrisk lrnprob1 read1
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
Residual Var.	88.856	13.083	67.751	118.695	---	---	5042.847
Coefficients:							
Intercept	87.848	1.321	85.187	90.382	4419.067	0.000	653.964
read1	0.500	0.047	0.409	0.593	114.659	0.000	4018.910
lrnprob1	-0.371	0.090	-0.550	-0.196	17.085	0.000	3402.882
atrisk	-1.934	1.871	-5.669	1.631	1.080	0.299	6822.952
read1*lrnprob1	0.012	0.005	0.003	0.022	6.855	0.009	3219.685
Standardized Coefficients:							
read1	0.671	0.046	0.573	0.753	212.802	0.000	3846.703
lrnprob1	-0.265	0.062	-0.384	-0.142	18.272	0.000	3280.798
atrisk	-0.061	0.059	-0.177	0.052	1.092	0.296	6854.792
read1*lrnprob1	0.170	0.060	0.047	0.281	7.794	0.005	3919.842
Proportion Variance Explained							
by Coefficients	0.610	0.050	0.500	0.698	---	---	4529.480
by Residual Variation	0.390	0.050	0.302	0.500	---	---	4529.480

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue.

The lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of first grade reading scores ($\beta_1 = 0.50$) is the effect of that predictor at the mean of the first-grade learning problems, and the learning problems slope ($\beta_2 = -0.37$) similarly reflects the conditional effect at the reading score mean. The interaction slope captures the change in the first-grade reading slope for each one-unit increase in learning problems (and vice versa). Specifically, the positive coefficient ($\beta_3 = 0.012$) indicates that the association between first and ninth grade reading scores becomes stronger (i.e., more positive) as learning problems increase. That is, the predictive power of early reading on later reading is strongest for students with elevated learning problem ratings in first grade. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion.

The SIMPLE command prints a table of conditional effects (simple slopes) of READ1 at different standard deviation units of LRNPROB1. The output is shown below. Consistent with the positive interaction coefficient, the simple slopes increase in strength as learning problems ratings increase (and vice versa). All the tabled conditional effects are statistically significant at $p < .05$ because the null value does not fall within the 95% credible intervals. The frequentist test statistics and p-values give the same conclusion.

Conditional Effects	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
read1 | lrnprob1 @ +2 SD							
Intercept	79.745	2.657	74.354	84.811	900.174	0.000	1199.336
Slope	0.767	0.119	0.540	1.007	42.054	0.000	2894.549
read1 | lrnprob1 @ +1 SD							
Intercept	83.791	1.833	80.097	87.310	2088.473	0.000	872.529
Slope	0.634	0.074	0.493	0.784	74.314	0.000	2879.088
read1 | lrnprob1 @ 0							
Intercept	87.848	1.321	85.187	90.382	4419.067	0.000	653.964
Slope	0.500	0.047	0.409	0.593	114.659	0.000	4018.910

```
read1 | lrnprob1 @ -1 SD
  Intercept                      91.872      1.488     89.002     94.868   3812.774      0.000   1250.238
  Slope                           0.368      0.066      0.234      0.494     30.896      0.000   5267.720

read1 | lrnprob1 @ -2 SD
  Intercept                      95.915      2.183     91.693    100.327   1931.435      0.000   3009.201
  Slope                           0.234      0.109      0.018      0.439      4.538      0.033   4262.126

                              ------------------------------------------------------------------------------

                              NOTE: Intercepts are computed by setting all predictors
                                    not involved in the conditional effect to zero.
```

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

10.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex10.2.imp` is identical `Ex10.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword

saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
    read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex10.R

```
1  library(rblimp)
2  load('behaviorachievement.rda')
3
```

```
mymodel <- rblimp(
  data = behaviorachievement,
  ordinal = 'atrisk',
  center = 'read1 lrnprob1 atrisk',
  model = '
    focal.model:
    read9 ~ read1 lrnprob1 read1*lrnprob1 atrisk ;
    auxiliary.models:
    stanread7 read2  ~ read9 read1 lrnprob1 atrisk',
  simple = 'read1 | lrnprob1',
  seed = 90291,
  burn = 5000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The SAVE command is no longer necessary because imputations are automatically stored in a rblimp object called mymodel@imputations. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

10.5 Analyzing Multiple Imputations in R

Continuing with the previous rblimp script, the following excerpt from Ex10.R shows how to perform multiple imputation inference. The script requires the mitml package (Grund et al., 2023).

R Script Ex10.R

```
library(rblimp)
library(mitml)
load('behaviorachievement.rda')

mymodel <- rblimp(...)

implist <- as.mitml(mymodel)

```

```
9   mean_read1 <- mean(unlist(lapply(implist, function(df) mean(df$read1))))
10  mean_lrnprob1 <- mean(unlist(lapply(implist, function(df) mean(df$lrnprob1))))
11  mean_atrisk <- mean(unlist(lapply(implist, function(df) mean(df$atrisk))))
12  for (i in 1:length(implist)) {
13    implist[[i]]$read1.cgm <- implist[[i]]$read1 - mean_read1
14    implist[[i]]$lrnprob1.cgm <- implist[[i]]$lrnprob1 - mean_lrnprob1
15    implist[[i]]$atrisk.cgm <- implist[[i]]$atrisk - mean_atrisk
16  }
17
18  fit <- with(implist,
19    lm(read9 ~ read1.cgm + lrnprob1.cgm + read1.cgm:lrnprob1.cgm + atrisk.cgm))
20
21  estimates <- testEstimates(fit, extra.pars = T, df.com = 133)
22  estimates
23  confint(estimates)
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 9 through 16 center three predictors at their pooled grand means. Lines 18 and 19 fit the focal regression model using the `lm` function, and line 21 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 22 and 23 print the pooled estimates and confidence intervals.

Following a significant interaction effect, researchers typically examine the slope of the focal predictor at different values of the moderator. The final code block below computes these conditional effects or simple slopes of first-grade reading scores at the learning problem mean and at plus and minus one standard deviation from the mean.

R Script Ex10.R, continued

```
24  lrnprob1.sd <- mean(unlist(lapply(implist, (function(x) sd(x$lrnprob1.cgm)))))
25
26  slp_high <- 'read1.cgm + read1.cgm*lrnprob1.cgm*1*10.77'
27  testConstraints(fit, constraints = slp_high, df.com = 133)
28
```

```
29   slp_mean <- 'read1.cgm + read1.cgm*lrnprob1.cgm*0*10.77'
30   testConstraints(fit, constraints = slp_mean, df.com = 133)
31
32   slp_low <- 'read1.cgm + read1.cgm*lrnprob1.cgm*-1*10.77'
33   testConstraints(fit, constraints = slp_low, df.com = 133)
```

Line 24 computes the pooled standard deviation of the moderator. Line 20 prints the value, which equals 10.77. Lines 26, 29, and 32 are text strings that define the computation of the conditional effect of `READ1` at the mean of `LRNPROB1` and at plus and minus one standard deviation from the mean. Lines 27, 30, and 33 use the `testConstraints` function in `mitml` to compute the pooled coefficients and test statistics.

10.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                       Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)              87.916     0.871   100.979    97.473     0.000     0.182     0.171
read1.cgm                 0.499     0.045    11.041    90.714     0.000     0.220     0.198
lrnprob1.cgm             -0.372     0.086    -4.321    74.600     0.000     0.331     0.268
atrisk.cgm               -2.053     1.779    -1.154   121.933     0.251     0.056     0.068
read1.cgm:lrnprob1.cgm    0.012     0.005     2.547    63.202     0.013     0.437     0.325

                  Estimate
Residual~~Residual   88.260

Hypothesis test adjusted for small samples with df=[133]
complete-data degrees of freedom.
                             2.5 %     97.5 %
(Intercept)            86.18838273 89.6441345
```

```
read1.cgm                0.40947708  0.5891414
lrnprob1.cgm            -0.54303533 -0.2003056
atrisk.cgm              -5.57347427  1.4681129
read1.cgm:lrnprob1.cgm   0.00255251  0.0211242
```

The lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of first grade reading scores ($\hat{\beta}_1 = 0.50$) is the effect of that predictor at the mean of the first-grade learning problems, and the learning problems slope ($\hat{\beta}_2 = -0.37$) similarly reflects the conditional effect at the reading score mean. The interaction slope captures the change in the first-grade reading slope for each one-unit increase in learning problems (and vice versa). Specifically, the positive coefficient ($\hat{\beta}_3 = 0.012$) indicates that the association between first and ninth grade reading scores becomes stronger (i.e., more positive) as learning problems increase. That is, the predictive power of early reading on later reading is strongest for students with elevated learning problem ratings in first grade. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation. The output also includes a table with standardized coefficients and the *R*-squared statistic.

Finally, the printed output also includes the table of conditional effects. The output is shown below. Consistent with the positive interaction coefficient, the simple slopes increase in strength as learning problems ratings increase (and vice versa). All the tabled conditional effects are statistically significant at $p < .05$.

```
Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                                                 Estimate Std. Error
   read1.cgm + read1.cgm*lrnprob1.cgm*1*10.77:     -1.500      0.482

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
      9.685       1  75.693   0.003   0.345

Hypothesis test adjusted for small samples with df=[133]
complete-data degrees of freedom.
```

```
Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                                                 Estimate Std. Error
   read1.cgm + read1.cgm*lrnprob1.cgm*0*10.77:      0.499      0.045

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
    121.909       1  95.723   0.000   0.220

Hypothesis test adjusted for small samples with df=[133]
complete-data degrees of freedom.


Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                                                  Estimate Std. Error
   read1.cgm + read1.cgm*lrnprob1.cgm*-1*10.77:      2.499      0.514

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     23.660       1  76.459   0.000   0.339

Hypothesis test adjusted for small samples with df=[133]
complete-data degrees of freedom.
```

10.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). Section 6.7 shows the contents of this file.

The Mplus input file for analyzing the imputations is `Ex10.inp`.

Mplus Script Ex10.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
6     read1 read2 read3 read9 read9grp stanread7
7     math1 math2 math3 math9 math9grp stanmath7;
8   usevariables = read9 read1 lrnprob1 atrisk product;
9   DEFINE:
10  center read1 lrnprob1 atrisk (grandmean);
11  product = read1 * lrnprob1;
12  MODEL:
13  read9 on read1 lrnprob1 product atrisk (beta1-beta4);
14  MODEL CONSTRAINT:
15  new(lrnprobvar slp_low slp_mean slp_high);
16  lrnprobvar = 114.354;
17  slp_high = beta1 + beta3*1*sqrt(lrnprobvar);
18  slp_mean = beta1 + beta3*0*sqrt(lrnprobvar);
19  slp_low =  beta1 - beta3*1*sqrt(lrnprobvar);
19  OUTPUT:
20  stdyx cinterval;
```

The major commands are like those from previous examples (see Section 1.2). Consistent with previous multiple imputation analysis scripts, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Third, the `MODEL` section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. The code block below shows the analysis and pooling script.

The script also invokes several new features. On line 10, the `center` subcommand under the `DEFINE` command centers the two interacting predictors at their grand means, and line 11 computes a new variable equal to the product of the centered scores. Importantly, new variables computed with the `DEFINE` command must appear at the end of the `usevariables` list on line 8. Beginning on line 14, the `MODEL CONSTRAINT` command is used to compute conditional effects or

simple slopes. First, line 15 assigns names to four new parameters (the variance of the moderator and three simple slopes). Line 16 inputs the variance of the moderator (obtained from the descriptive statistics on the output), and lines 17 through 20 compute the conditional effect of READ1 at the mean of LRNPROB1 and at plus and minus one standard deviation from the mean.

10.8 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The Rate of Missing column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
    READ1            0.499      0.044     11.232      0.000      0.185
    LRNPROB1        -0.372      0.084     -4.420      0.000      0.247
    PRODUCT          0.012      0.005      2.580      0.010      0.319
    ATRISK          -2.053      1.748     -1.174      0.240      0.055

 Intercepts
    READ9           87.917      0.846    103.950      0.000      0.138

 Residual Variances
    READ9           85.062     11.810      7.203      0.000      0.250
```

The lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of first grade reading scores ($\hat{\beta}_1 = 0.50$) is the effect of that predictor at the mean of the first-grade learning problems, and the learning problems slope ($\hat{\beta}_2 = -0.37$) similarly reflects the conditional effect at the reading score mean. The interaction slope captures the change in the first-grade reading slope for each one-unit increase in learning problems (and vice versa). Specifically, the positive coefficient ($\hat{\beta}_3 = 0.012$) indicates that the association between first and ninth grade reading scores becomes stronger (i.e.,

more positive) as learning problems increase. That is, the predictive power of early reading on later reading is strongest for students with elevated learning problem ratings in first grade. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation. The output also includes a table with standardized coefficients and the *R*-squared statistic.

Finally, the printed output also includes the table of conditional effects, which were computed using the `MODEL CONSTRAINT` command. The output is shown below. Consistent with the positive interaction coefficient, the simple slopes increase in strength as learning problems ratings increase (and vice versa). All the tabled conditional effects are statistically significant at $p < .05$.

```
                                                  Two-Tailed   Rate of
                    Estimate       S.E.  Est./S.E.    P-Value    Missing
...

New/Additional Parameters
    LRNPROBV         114.354      0.022   5129.998      0.000      1.000
    SLP_LOW            0.373      0.061      6.079      0.000      0.098
    SLP_MEAN           0.499      0.044     11.232      0.000      0.185
    SLP_HIGH           0.626      0.071      8.844      0.000      0.380
```

10.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex10.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex10.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3    /imputation_ id male hispanic riskgrp atrisk
4     behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
5     math1 math2 math3 math9 math9grp stanmath7.
```

```
6   EXE.
7
8   AGGREGATE
9     /outfile = * mode = addvariables overwrite = yes
10    /lrnprob1_mean = mean(lrnprob1)
11    /read1_mean = mean(read1)
12    /atrisk_mean = mean(atrisk).
13  EXE.
14
15  COMPUTE lrnprob1_cgm = lrnprob1 - lrnprob1_mean.
16  COMPUTE read1_cgm = read1 - read1_mean.
17  COMPUTE atrisk_cgm = atrisk - atrisk_mean.
18  COMPUTE lrnprob1_by_read1 = lrnprob1_cgm * read1_cgm.
19  EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 8, the `AGGREGATE` command adds the grand means to the data. Then, beginning on line 15, each variable is centered at its pooled grand mean.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 20 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 21 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 22.

SPSS Script Ex10.spwb, continued

```
20
21  SORT CASES by imputation_.
22  SPLIT FILE layered by imputation_.
23  REGRESSION
24    /descriptives mean stddev corr sig n
25    /dependent read9
26    /method enter read1_cgm lrnprob1_cgm lrnprob1_by_read1 atrisk_cgm.
```

10.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients		Standardized Coefficients			Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
			B	Std. Error	Beta	t	Sig.			
1.00	1	(Constant)	87.984	.786		111.946	.000			
		read1_cgm	.476	.040	.670	11.964	.000			
		lrnprob1_cgm	-.363	.074	-.266	-4.914	.000			
		lrnprob1_by_read1	.010	.004	.150	2.772	.006			
		atrisk_cgm	-2.337	1.701	-.077	-1.374	.172			
2.00	1	(Constant)	87.702	.801		109.468	.000			
		read1_cgm	.512	.041	.693	12.365	.000			
		lrnprob1_cgm	-.372	.075	-.270	-4.941	.000			
		lrnprob1_by_read1	.014	.004	.186	3.403	.001			
		atrisk_cgm	-1.340	1.726	-.044	-.776	.439			
• • •										
20.00	1	(Constant)	87.891	.788		111.539	.000			
		read1_cgm	.506	.041	.691	12.441	.000			
		lrnprob1_cgm	-.314	.074	-.230	-4.226	.000			
		lrnprob1_by_read1	.013	.004	.176	3.247	.001			
		atrisk_cgm	-2.160	1.688	-.071	-1.280	.203			
Pooled	1	(Constant)	87.916	.871		100.979	.000	.156	.182	.992
		read1_cgm	.499	.045		11.041	.000	.183	.220	.991
		lrnprob1_cgm	-.372	.086		-4.321	.000	.253	.331	.987
		lrnprob1_by_read1	.012	.005		2.547	.012	.311	.437	.985
		atrisk_cgm	-2.053	1.779		-1.154	.248	.053	.056	.997

a. Dependent Variable: read9

The lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of first grade reading scores ($\hat{\beta}_1$= 0.50) is the effect of that predictor at the mean of the first-grade learning problems, and the learning problems slope ($\hat{\beta}_2$ = −0.37) similarly reflects the conditional effect at the reading score mean. The interaction slope captures the change in the first-grade reading slope for each one-unit increase in learning problems (and vice versa). Specifically, the positive coefficient ($\hat{\beta}_3$= 0.012) indicates that the association between first and ninth grade reading scores becomes stronger (i.e.,

more positive) as learning problems increase. That is, the predictive power of early reading on later reading is strongest for students with elevated learning problem ratings in first grade. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation. The output also includes a table with standardized coefficients and the R-squared statistic. Finally, the printed output also includes the table of conditional effects. The output is shown below. Consistent with the positive interaction coefficient, the simple slopes increase in strength as learning problems ratings increase (and vice versa). All of the tabled conditional effects are statistically significant at $p < .05$.

11

MCMC: Curvilinear Regression

This example illustrates a multiple regression analysis with an incomplete curvilinear effect. The analysis uses the `mathachievement.dat` data set taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
MATHPOST	Math achievement posttest	18.0	Numeric
ANXIETY	Math anxiety composite	8.4	Numeric
FRLUNCH	Lunch assistance code	4.4	0 = None, 1 = Free/reduced lunch
EFFICACY	Math self-efficacy rating	9.6	Ordinal (1 to 6)
MATHPRE	Math achievement pretest	0	Numeric
	Auxiliary Variables		
ATRISK	Behavioral disorder risk	5.2	0 = Low risk, 1 = At-risk
STANREAD	Standardized reading	9.2	Numeric

11.1 Analysis Model

The analysis model features math posttest scores regressed on anxiety and its square, the lunch assistance dummy code, math self-efficacy ratings, and math pretest scores.

$$\begin{aligned} MATHPOST &= \beta_0 + \beta_1(ANXIETY) + \beta_2(ANXIETY^2) \\ &+ \beta_3(FRLUNCH) + \beta_4(EFFICACY) + \beta_5(MATHPRE) + \varepsilon \end{aligned} \tag{24}$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Curvilinear regression models (and models with non-linearities more generally) require a factored regression specification that assigns separate distributions to the predictors and outcome. By default, Blimp invokes a multivariate normal distribution for numeric predictors and latent response scores.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} ATRISK^* &= \gamma_{03} + \gamma_{13}(MATHPOST) + \gamma_{23}(ANXIETY) \\ &+ \gamma_{33}(FRLUNCH) + \gamma_{43}(EFFICACY) + \gamma_{53}(MATHPRE) + \epsilon_3 \\ STANREAD &= \gamma_{04} + \gamma_{14}(ATRISK) + \gamma_{24}(MATHPOST) + \gamma_{34}(ANXIETY) \\ &+ \gamma_{44}(FRLUNCH) + \gamma_{54}(EFFICACY) + \gamma_{64}(MATHPRE) + \epsilon_4 \end{aligned} \tag{25}$$

The `ATRISK` model is a probit regression, with the binary outcome model as a latent response variable (denoted by the asterisk superscript). Again, the entire collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

11.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex11.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex11.1.imp

```
1  DATA: mathachievement.dat;
2  VARIABLES: id condition male frlunch atrisk stanread efficacy anxiety
```

```
3      mathpre mathpost;
4   ORDINAL: frlunch atrisk efficacy;
5   MISSING: 999;
6   FIXED: mathpre;
7   CENTER: anxiety;
8   MODEL:
9   focal.model:
10  mathpost ~ anxiety anxiety^2@beta2 frlunch mathpre efficacy;
11  auxiliary.models:
12  stanread atrisk ~ mathpost anxiety frlunch efficacy mathpre;
13  SEED: 90291;
14  BURN: 10000;
15  ITERATIONS: 10000;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the `VARIABLES` command names the data columns. The `ORDINAL` command on line 4 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The `MISSING` command on line 5 defines a global missing value code as 999.

The `FIXED`, `CENTER`, and `MODEL` blocks can be viewed as a set. The `FIXED` command identifies a complete predictor, which does not require a distribution or regression model. The `CENTER` command deviates anxiety scores (the variable with the non-linear term) at their iteratively-estimated grand mean. Beginning on line 8, the `MODEL` command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses labels (`focal.model` and `auxiliary.models`) to order output tables, such that the focal model appears first followed by the auxiliary variable models. The focal model listed on line 10 includes a squared term, which is specified by appending `^2` to the variable name. The quadratic slope coefficient is labeled using the @ symbol. Blimp automatically configures the explanatory variable models under the assumption that they are normally distributed. Line 12 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 25; in the first model, `READ2` is regressed on the focal variables, and the second model features `STANREAD7` regressed on `READ2` and the focal variables.

Finally, lines 13 through 15 can be viewed as a block of commands that specify features of the MCMC algorithm: the `SEED` command gives an integer string that initializes the random number

generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex11.R is shown below.

rblimp Script Ex11.R

```
library(rblimp)
load('mathachievement.rda')

mymodel <- rblimp(
  data = mathachievement,
  ordinal = 'atrisk frlunch efficacy',
  fixed = 'mathpre',
  center = 'anxiety',
  model = '
    focal.model:
    mathpost ~ anxiety anxiety^2@beta2 frlunch efficacy mathpre;
    auxiliary.models:
    stanread atrisk ~ mathpost anxiety frlunch efficacy mathpre',
  seed = 90291,
  burn = 10000,
  iter = 10000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script. Finally, the output(mymodel) function prints the Blimp output.

11.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          251 to 500             1.045            61
                          501 to 1000            1.087            58
                          751 to 1500            1.042            58
                                  ...              ...            ..
                         4501 to 9000            1.045            60
                         4751 to 9500            1.009            58
                         5001 to 10000           1.012            59
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3.

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior

medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  mathpost

Grand Mean Centered: anxiety
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
Residual Var.	50.966	5.388	41.841	63.021	---	---	5413.246
Coefficients:							
Intercept	32.689	3.457	25.967	39.568	89.528	0.000	6410.684
anxiety	0.041	0.084	-0.122	0.206	0.237	0.626	4221.113
frlunch	-5.840	1.073	-7.949	-3.750	29.724	0.000	4993.497
efficacy	1.103	0.342	0.444	1.781	10.449	0.001	5838.380
mathpre	0.471	0.067	0.338	0.602	49.615	0.000	6319.544
anxiety^2	-0.021	0.006	-0.033	-0.009	11.705	0.001	5091.389
Standardized Coefficients:							
anxiety	0.031	0.064	-0.093	0.157	0.238	0.626	4228.247
frlunch	-0.298	0.051	-0.394	-0.195	33.903	0.000	5053.969
efficacy	0.184	0.056	0.074	0.292	10.798	0.001	5698.632
mathpre	0.423	0.055	0.309	0.523	59.287	0.000	6522.033
anxiety^2	-0.204	0.058	-0.315	-0.087	12.120	0.000	5054.272
Proportion Variance Explained							
by Coefficients	0.453	0.045	0.359	0.539	---	---	5780.687
by Residual Variation	0.547	0.045	0.461	0.641	---	---	5780.687

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue.

In a curvilinear regression model, the lower-order term for math anxiety is a conditional effect that depends on scaling or centering. The slope conveys the instantaneous linear change in the outcome at the anxiety mean, controlling for all other predictors ($\beta_1 = 0.04$). The negative quadratic coefficient ($\beta_2 = -0.02$) indicates that the positive association at the mean decreases (i.e., becomes less positive) as anxiety increases (and vice versa). At high enough levels of anxiety, the association becomes negative, such that anxiety has a debilitating effect on math performance. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in OUTCOME MODEL ESTIMATES section with the focal results, and the auto-generated predictor models are displayed under the heading PREDICTOR MODEL ESTIMATES. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

11.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file Ex11.2.imp is identical Ex11.1.imp, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     id condition male frlunch atrisk stanread efficacy anxiety
     mathpre mathpost

   stacked = './imps/imps.dat'

     imp# id condition male frlunch atrisk stanread efficacy
     anxiety mathpre mathpost
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In rblimp, the NIMPS and CHAINS commands are added as input parameters to the function as follows.

rblimp Script Ex11.R

```
library(rblimp)
load('mathachievement.rda')

mymodel <- rblimp(
  data = mathachievement,
  ordinal = 'atrisk frlunch efficacy',
  fixed = 'mathpre',
  center = 'anxiety',
  model = '
     focal.model:
     mathpost ~ anxiety anxiety^2@beta2 frlunch efficacy mathpre;
     auxiliary.models:
     stanread atrisk ~ mathpost anxiety frlunch efficacy mathpre',
  seed = 90291,
  burn = 10000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

11.5 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex11.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex11.R

```
library(rblimp)
library(mitml)
load('mathachievement.rda')
```

```
4
5   mymodel <- rblimp(...)
6
7   implist <- as.mitml(mymodel)
8
9   mean_anxiety <- mean(unlist(lapply(implist, function(df) mean(df$anxiety))))
10  for (i in 1:length(implist)) {
11    implist[[i]]$anxiety.cgm <- implist[[i]]$anxiety - mean_anxiety
12  }
13
14  fit <- with(implist,
15    lm(mathpost ~ anxiety.cgm + I(anxiety.cgm^2) + frlunch  + efficacy + mathpre))
16  estimates <- testEstimates(fit, extra.pars = T, df.com = 244)
17  estimates
18  confint(estimates)
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 9 through 12 center the focal predictor at its grand mean. Lines 14 and 15 fit the focal regression model using the `lm` function, and line 16 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 17 and 18 print the pooled estimates and confidence intervals.

11.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    32.943     3.318     9.929   177.001     0.000     0.150     0.140
anxiety.cgm     0.034     0.084     0.404    96.483     0.687     0.415     0.307
anxiety.sq     -0.021     0.006    -3.616   147.802     0.000     0.221     0.192
frlunch        -5.687     1.147    -4.960    76.330     0.000     0.554     0.373
efficacy        1.052     0.344     3.058   107.170     0.003     0.361     0.279
mathpre         0.471     0.064     7.336   173.143     0.000     0.159     0.147


                   Estimate
Residual~~Residual   50.358

Hypothesis test adjusted for small samples with df=[244]
complete-data degrees of freedom.

                  2.5 %       97.5 %
(Intercept) 26.39566909 39.490858174
anxiety.cgm -0.13268668  0.200544222
anxiety.sq  -0.03287417 -0.009639516
frlunch     -7.97089460 -3.403528840
efficacy     0.37008949  1.734175237
mathpre      0.34392328  0.597097949
```

In a curvilinear regression model, the lower-order term for math anxiety is a conditional effect that depends on scaling or centering. The slope conveys the instantaneous linear change in the outcome at the anxiety mean, controlling for all other predictors ($\hat{\beta}_1 = 0.03$). The negative quadratic coefficient ($\hat{\beta}_2 = -0.02$) indicates that the positive association at the mean decreases (i.e., becomes less positive) as anxiety increases (and vice versa). At high enough levels of anxiety, the association becomes negative, such that anxiety has a debilitating effect on math performance. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

11.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data

set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). Section 6.7 shows the contents of this file.

The Mplus input file for analyzing the imputations is `Ex11.inp`.

Mplus Script Ex11.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = id condition male frlunch atrisk stanread
6     efficacy anxiety mathpre mathpost;
7   usevariables = mathpost anxiety frlunch efficacy mathpre anxietysq;
8   DEFINE:
9   center anxiety (grandmean);
10  anxietysq = anxiety^2;
11  MODEL:
12  mathpost on anxiety anxietysq frlunch efficacy mathpre;
13  OUTPUT:
14  stdyx cinterval;
```

The major commands are described in previous examples. Consistent with previous multiple imputation analysis scripts, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Finally, the `MODEL` section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. The script also invokes one new feature. On line 9, the `center` subcommand under the `DEFINE` command centers anxiety scores at their grand mean. Line 10 then computes a new variable equal to the square of the centered predictor. Importantly, new variables computed with the `DEFINE` command must appear at the end of the `usevariables` list on line 7. The script is shown below.

11.8 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS
```

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value	Rate of Missing
MATHPOST ON					
ANXIETY	0.034	0.083	0.407	0.684	0.304
ANXIETYSQ	-0.021	0.006	-3.652	0.000	0.187
FRLUNCH	-5.687	1.138	-4.998	0.000	0.371
EFFICACY	1.052	0.341	3.085	0.002	0.276
MATHPRE	0.471	0.063	7.414	0.000	0.142
Intercepts					
MATHPOST	32.944	3.283	10.035	0.000	0.135
Residual Variances					
MATHPOST	49.149	5.588	8.795	0.000	0.388

In a curvilinear regression model, the lower-order term for math anxiety is a conditional effect that depends on scaling or centering. The slope conveys the instantaneous linear change in the outcome at the anxiety mean, controlling for all other predictors ($\hat{\beta}_1 = 0.03$). The negative quadratic coefficient ($\hat{\beta}_2 = -0.02$) indicates that the positive association at the mean decreases (i.e., becomes less positive) as anxiety increases (and vice versa). At high enough levels of anxiety, the association becomes negative, such that anxiety has a debilitating effect on math performance. The output also includes a table with standardized coefficients and the *R*-squared statistic. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

11.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the stacked keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is Ex11.spwb. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex11.spwb

```
1   CD '/users/username/desktop'.
2   DATA LIST free file =  'imps.dat'
3     /imputation_ id condition male frlunch atrisk stanread
4      efficacy anxiety mathpre mathpost.
5   EXE.
6
7   AGGREGATE
8     /outfile = * mode = addvariables overwrite = yes
9     /anxiety_mean = mean(anxiety).
10  EXE.
11
12  COMPUTE anxiety_cgm = anxiety - anxiety_mean.
13  COMPUTE anxiety_sq = anxiety_cgm**2.
14  EXE.
```

The first line uses the CD command to change the working directory to the desktop. The username portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 7, the AGGREGATE command adds the grand means to the data. Then, beginning on line 12, a centered version of the focal predictor is computed along with its square.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 15 sorts the data by the imputation index variable, and the SPLIT

FILE command on line 16 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 17.

SPSS Script Ex11.spwb, continued

```
15  SORT CASES by imputation_.
16  SPLIT FILE layered by imputation_.
17  REGRESSION
18    /descriptives mean stddev corr sig n
19    /dependent mathpost
20    /method enter anxiety_cgm anxiety_sq frlunch efficacy mathpre.
```

11.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

In a curvilinear regression model, the lower-order term for math anxiety is a conditional effect that depends on scaling or centering. The slope conveys the instantaneous linear change in the outcome at the anxiety mean, controlling for all other predictors ($\hat{\beta}_1 = 0.03$). The negative quadratic coefficient ($\hat{\beta}_2 = -0.02$) indicates that the positive association at the mean decreases (i.e., becomes less positive) as anxiety increases (and vice versa). At high enough levels of anxiety, the association becomes negative, such that anxiety has a debilitating effect on math performance. Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	1	(Constant)	33.999	3.159		10.763	<.001			
		anxiety_cgm	-.070	.072	-.055	-.985	.326			
		anxiety_sq	-.017	.005	-.164	-3.050	.003			
		frlunch	-5.030	.947	-.264	-5.310	<.001			
		efficacy	.650	.303	.112	2.142	.033			
		mathpre	.470	.061	.434	7.717	<.001			
2.00	1	(Constant)	33.888	3.387		10.007	<.001			
		anxiety_cgm	.055	.077	.041	.715	.475			
		anxiety_sq	-.021	.006	-.190	-3.510	<.001			
		frlunch	-7.040	1.010	-.345	-6.971	<.001			
		efficacy	1.038	.325	.166	3.189	.002			
		mathpre	.454	.065	.392	6.941	<.001			
• • •										
20.00	1	(Constant)	31.923	3.099		10.300	<.001			
		anxiety_cgm	-.003	.070	-.002	-.044	.965			
		anxiety_sq	-.022	.005	-.224	-4.285	<.001			
		frlunch	-5.500	.907	-.283	-6.063	<.001			
		efficacy	.890	.293	.149	3.041	.003			
		mathpre	.509	.059	.460	8.597	<.001			
Pooled	1	(Constant)	32.943	3.318		9.929	<.001	.132	.150	.993
		anxiety_cgm	.034	.084		.404	.686	.299	.415	.985
		anxiety_sq	-.021	.006		-3.616	<.001	.184	.221	.991
		frlunch	-5.687	1.147		-4.960	<.001	.365	.554	.982
		efficacy	1.052	.344		3.058	.002	.271	.361	.987
		mathpre	.471	.064		7.336	<.001	.139	.159	.993

a. Dependent Variable: mathpost

12

FCS Multiple Imputation: Paired-Samples Comparison

This example illustrates model-agnostic fully conditional specification multiple imputation for a paired-samples test involving pretest and posttest scores. The analysis uses the `mathachievement.dat` data set taken from an educational intervention where 250 students were assigned to an intervention and comparison condition. The file includes pretest and posttest math achievement scores, a measure of math self-efficacy, standardized reading scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
MATHPRE	Math achievement pretest	0	Numeric
MATHPOST	Math achievement posttest	18.0	Numeric
	Auxiliary Variables		
FRLUNCH	Lunch assistance code	4.4	0 = None, 1 = Free/reduced lunch
STANREAD	Standardized reading	9.2	Numeric
EFFICACY	Math self-efficacy rating	9.6	Ordinal (1 to 6)

12.1 Imputation and Analysis Models

A common goal of model-agnostic imputation is to generate imputations for different purposes (e.g., descriptive summaries, several analyses within the same project). To illustrate multiple imputation with the fully conditional specification algorithm (i.e., multiple imputation by chained equations, or MICE; van Buuren, 2018), suppose that one use of the filled-in data sets involves a paired-samples test of the changes between pretest and posttest. The analysis can be cast as an empty regression model with change scores as the outcome variable.

$$CHANGE = \beta_0 + \varepsilon \tag{26}$$

The variable CHANGE is computed as MATHPOST minus MATHPRE.

Fully conditional specification uses a sequence of regression models to fill in missing values. Specifically, each MCMC iteration fits a series of models where one incomplete variable is regressed on all other variables. The predicted values and residual variance from each model define the center and spread of the imputed values, which are drawn at random from a normal distribution. After imputing the missing scores, the filled-in variable becomes a predictor in all other imputation models in the sequence. The imputation stage should include all variables and effects for the subsequent analyses, and it should incorporate auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). The imputation models use the two math scores and three auxiliary variables. Difference scores are computed from the imputed data prior to analysis.

12.2 Blimp and rblimp FCS Scripts

The code block below shows Blimp script Ex12.inp. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex12.imp

```
DATA: mathachievement.dat;
VARIABLES: id condition male frlunch atrisk stanread efficacy anxiety
   mathpre mathpost;
ORDINAL: frlunch efficacy;
MISSING: 999;
FIXED: mathpre;
FCS: mathpost mathpre frlunch stanread efficacy;
SEED: 90291;
BURN: 5000;
ITERATIONS: 10000;
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The ORDINAL command on line 4 identifies a pair of binary variables. Binary variables can alternatively be identified using the NOMINAL command because the underlying statistical models are identical. Finally, the MISSING command on line 5 defines a global missing value code as 999.

Next, the FCS command lists all variables—complete or incomplete—that are included in the imputation phase. The FIXED command identifies a complete variable that does not require imputation. This reduces computational time because complete variables do not require a regression model. Lines 8 through 10 can also be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the imputation model summaries are based (essentially, the total number of MCMC cycles across all chains following the warm-up period).

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

The corresponding rblimp script Ex12.R is shown below.

rblimp Script Ex12.R

```
1   library(rblimp)
2   load('mathachievement.rda')
3
4   mymodel <- rblimp_fcs(
5     data = mathachievement,
```

```
6     ordinal = 'frlunch efficacy',
7     fixed = 'mathpre',
8     variables = 'mathpost mathpre frlunch stanread efficacy',
9     seed = 90291,
10    burn = 5000,
11    iter = 10000,
12    nimps = 20,
13    chains = 20)
14    output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp_fcs` function. The two exceptions are the `MISSING` and `FCS` commands. The former is omitted because that information is contained in the R data file. The `FCS` command is replaced by a `variables` parameter that lists the variables to be included in the imputation model. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like variable lists are enclosed in quotes, and numeric inputs like the seed and number of iterations do not require quotes. Finally, the `output(mymodel)` function prints the Blimp output.

12.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          126 to 250             1.480            23
                          251 to 500             1.395            24
```

```
 376 to 750            1.272          23
        ...              ...          ..
2251 to 4500           1.048          22
2376 to 4750           1.038          23
2501 to 5000           1.042          23
```

The next output excerpt shows information about the data and the variables in the imputation models.

```
DATA INFORMATION:

  Sample Size:              250
  Missing Data Rates:

                frlunch = 04.40
               stanread = 09.20
               efficacy = 09.60
               mathpost = 18.00

VARIABLES IN IMPUTATION MODEL:

  Fixed variables:        mathpre
  Incomplete continuous:  stanread mathpost
  Incomplete ordinal:     frlunch efficacy

  NUMBER OF PARAMETERS
 Imputation Models:      26
```

MCMC estimation produces a distribution for each parameter in every unique imputation model. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. The Blimp output includes tables of regression parameters for every incomplete variable's imputation

model. The imputation model parameters are not of substantive interest and would not be reported. An example table is shown below.

```
Missing variable:   mathpost

Parameters                      Median     StdDev       2.5%      97.5%        PSR      N_Eff
                       ---------------------------------------------------------------------

Grand Mean                      56.504      0.555     55.397     57.567      1.005   4353.468

Level 1:
  frlunch                       -2.188      0.575     -3.270     -1.038      1.006   3351.406
  stanread                       0.189      0.060      0.073      0.306      1.004   4430.891
  efficacy                       1.222      0.517      0.216      2.252      1.005   4406.012
  mathpre                        0.476      0.062      0.354      0.596      1.002   6043.712
  Residual Var.                 45.242      4.994     36.454     56.257      1.004   4585.610

                       ---------------------------------------------------------------------
```

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     id condition male frlunch atrisk stanread efficacy anxiety
     mathpre mathpost

   stacked = './imps/imps.dat'

     imp# id condition male frlunch atrisk stanread efficacy
     anxiety mathpre mathpost
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does

not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

12.4 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex12.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex12.R

```
1   library(rblimp)
2   library(mitml)
3   load('mathachievement.rda')
4
5   mymodel <- rblimp_fcs(...)
6
7   implist <- as.mitml(mymodel)
8
9   for (i in 1:length(implist)) {
10    implist[[i]]$change <- implist[[i]]$mathpost -  implist[[i]]$mathpre
11  }
12
13  fit <- with(implist, lm(change ~ 1))
14  estimates <- testEstimates(fit, extra.pars = T, df.com = 249)
15  estimates
16  confint(estimates)
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 9 through 12 compute the change scores. Line 13 fits the focal regression model using the `lm` function, and line 14 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 15 and 16 print the pooled estimates and confidence intervals.

12.5 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)     6.418     0.578    11.112   181.675     0.000     0.147     0.137

                  Estimate
Residual~~Residual   72.732

Hypothesis test adjusted for small samples with df=[249]
complete-data degrees of freedom.

               2.5 %  97.5 %
(Intercept) 5.278415 7.55776
```

The results are interpreted in the same way as a complete-data paired-samples test. For example, the intercept represents the mean change from pretest to posttest. The corresponding test statistic indicates that the change is statistically different from zero ($t = 11.11$, $p < .001$).

12.6 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). As a reminder, the contents of this file are as follows.

```
imp1.dat
imp2.dat
imp3.dat
imp4.dat
imp5.dat
...
imp16.dat
imp17.dat
imp18.dat
imp19.dat
imp20.dat
```

The Mplus input file for analyzing the imputations is `Ex12.inp`.

Mplus Script Ex12.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = id condition male frlunch lowach stanread efficacy
6     anxiety mathpre mathpost;
7   usevariables =  change;
8   DEFINE:
9   change = mathpost - mathpre;
10  MODEL:
11  change;
12  OUTPUT:
13  cinterval;
```

Following previous imputation analysis examples, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. The `usevariables` subcommand of the `VARIABLE` command selects variables for the analysis. The `DEFINE` command beginning on line 8 computes the change or difference score by subtracting the pretest from posttest. Importantly, new variables computed with the `DEFINE` command must appear at the end of the `usevariables` list on line 7. In this example, the new

change score is the only variable in the model. Listing the change score variable in the MODEL section estimates the mean and variance of the variable. Finally, listing the cinterval keyword after OPTION prints confidence intervals. The code block below shows the analysis and pooling script.

12.7 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The Rate of Missing column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

 Means
    CHANGE           6.418      0.577     11.131      0.000      0.130

 Variances
    CHANGE          72.439      6.864     10.554      0.000      0.109
```

The results are interpreted in the same way as a complete-data paired-samples test. For example, the intercept represents the mean change from pretest to posttest. The corresponding test statistic indicates that the change is statistically different from zero ($z = 11.13$, $p < .001$).

12.8 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the stacked keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is Ex12.spwb. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex12.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3    /imputation_ id condition male frlunch atrisk stanread
4     efficacy anxiety mathpre mathpost.
5  EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 6 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 7 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 8.

SPSS Script Ex12.spwb, continued

```
6  SORT CASES by imputation_.
7  SPLIT FILE layered by imputation_.
8  T-TEST pairs = mathpost with mathpre (paired).
```

12.9 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the

fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

			Paired Differences							Significance				
						95% Confidence Interval of the Difference								
imputation_			Mean	Std. Deviation	Std. Error Mean	Lower	Upper	t	df	One-Sided p	Two-Sided p	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	Pair 1	mathpost - mathpre	6.65219	8.56047	.54141	5.58586	7.71852	12.287	249	.000	.000			
2.00	Pair 1	mathpost - mathpre	6.50856	8.31642	.52598	5.47263	7.54449	12.374	249	.000	.000			
...														
20.00	Pair 1	mathpost - mathpre	6.28921	8.58376	.54288	5.21998	7.35844	11.585	249	.000	.000			
Pooled	Pair 1	mathpost - mathpre	6.41809		.57760	5.28482	7.55135	11.112	759		.000	.129	.147	.994

The results are interpreted in the same way as a complete-data paired-samples test. For example, the intercept represents the mean change from pretest to post test. The corresponding test statistic indicates that the change is statistically different from zero ($t = 11.11$, $p < .001$).

13

FCS Multiple Imputation: Multiple Regression

This example illustrates model-agnostic fully conditional specification multiple imputation with multivariate normal data. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

13.1 Imputation and Analysis Models

A common goal of model-agnostic imputation is to generate imputations for different purposes (e.g., descriptive summaries, several analyses within the same project). To illustrate multiple imputation with the fully conditional specification algorithm (i.e., multiple imputation by

chained equations, or MICE; van Buuren, 2018), suppose that one use of the filled-in data sets involves a model where ninth grade broad reading scores are regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) + \varepsilon \tag{27}$$

Chapters 1 and 6 used the same analysis model to illustrate maximum likelihood estimation, MCMC estimation, and model-based multiple imputation.

Fully conditional specification uses a sequence of regression models to fill in missing values. Specifically, each MCMC iteration fits a series of models where one incomplete variable is regressed on all other variables. The predicted values and residual variance from each model define the center and spread of the imputed values, which are drawn at random from a normal distribution. After imputing the missing scores, the filled-in variable becomes a predictor in all other imputation models in the sequence. The imputation stage should include all variables and effects for the subsequent analyses, and it should incorporate auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). The imputation models use the four analysis variables and three auxiliary variables.

13.2 Blimp and rblimp FCS Scripts

The code block below shows Blimp script `Ex13.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex13.imp

```
DATA: behaviorachievement.dat;
VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
   read1 read2 read3 read9 read9grp stanread7
   math1 math2 math3 math9 math9grp stanmath7;
MISSING: 999;
FCS: read9 read1 lrnprob1 behsymp1 stanread7 read2;
SEED: 90291;
BURN: 2000;
ITERATIONS: 10000;
NIMPS: 20;
CHAINS: 20;
```

```
12  SAVE:
13  stacked = ./imps/imps.dat;
14  separate = ./imps/imp*.dat;
```

The first five lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns, and the MISSING command on line 5 defines a global missing value code as 999.

Next, the FCS command lists all variables—complete or incomplete—that are included in the imputation phase. Using the FIXED command to identify complete variables reduces computational time because these variables do not require a regression model. Lines 7 through 9 can also be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the imputation model summaries are based (essentially, the total number of MCMC cycles across all chains following the warm-up period).

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

The corresponding rblimp script Ex13.R is shown below.

rblimp Script Ex13.R

```
1  library(rblimp)
2  load('behaviorachievement.rda')
3
```

```
4   mymodel <- rblimp_fcs(
5     data = behaviorachievement,
6     variables = 'read9 read1 lrnprob1 behsymp1 stanread7 read2',
7     seed = 90291,
8     burn = 2000,
9     iter = 10000,
10    nimps = 20,
11    chains = 20)
12    output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp_fcs` function. The two exceptions are the `MISSING` and `FCS` commands. The former is omitted because that information is contained in the R data file. The `FCS` command is replaced by a `variables` parameter that lists the variables to be included in the imputation model. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like variable lists are enclosed in quotes, and numeric inputs like the seed and number of iterations do not require quotes. Finally, the `output(mymodel)` function prints the Blimp output.

13.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           51 to 100             1.436            22
                          101 to 200             1.245            22
```

```
151 to 300          1.132          22
       ...            ...          ..
901 to 1800         1.031          22
951 to 1900         1.024          22
1001 to 2000        1.022          22
```

The next output excerpt shows information about the data and the variables in the imputation models.

```
DATA INFORMATION:

  Sample Size:              138
  Missing Data Rates:

               behsymp1 = 03.62
               lrnprob1 = 02.17
                  read1 = 06.52
                  read2 = 09.42
                  read9 = 17.39
              stanread7 = 19.57

VARIABLES IN IMPUTATION MODEL:

  Incomplete continuous:  behsymp1 lrnprob1 read1 read2 read9 stanread7

  NUMBER OF PARAMETERS
 Imputation Models:      42
```

MCMC estimation produces a distribution for each parameter in every unique imputation model. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. The Blimp output includes tables of regression parameters for every incomplete variable's imputation

model. The imputation model parameters are not of substantive interest and would not be reported. An example table is shown below.

```
Missing variable:   behsymp1

Parameters                      Median     StdDev       2.5%      97.5%        PSR      N_Eff
                       ---------------------------------------------------------------------

Grand Mean                      49.506      1.093     47.355     51.590      1.006   2075.694

Level 1:
  lrnprob1                       0.731      0.071      0.591      0.872      1.002   8316.690
  read1                         -0.274      0.077     -0.422     -0.121      1.002   8393.286
  read2                          0.590      0.103      0.386      0.792      1.002   7435.432
  read9                         -0.457      0.102     -0.657     -0.254      1.003   8587.332
  stanread7                     -0.018      0.016     -0.048      0.014      1.003   6881.275
  Residual Var.                 55.104      7.569     42.773     72.497      1.003   7269.684

                       ---------------------------------------------------------------------
```

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
    read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does

not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

13.4 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex13.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex13.R

```
1   library(rblimp)
2   library(mitml)
3   load('behaviorachievement.rda')
4
5   mymodel <- rblimp_fcs(...)
6
7   implist <- as.mitml(mymodel)
8
9   fit <- with(implist, lm(read9 ~ read1 + lrnprob1 + behsymp1))
10  estimates <- testEstimates(fit, extra.pars = T, df.com = 134)
11  estimates
12  confint(estimates)
13
14  null <- with(implist, lm(read9 ~ 1))
15  testModels(fit, null, df.com = 134, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Line 9 fits the focal regression model using the `lm` function, and line 10 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 15 and 16 print the pooled estimates and confidence intervals. Finally, lines 14 and 14 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 14 that represents the null hypothesis, which in this case is

an empty regression model with just an intercept. On line 15, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

13.5 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
  Final parameter estimates and inferences obtained from 20 imputed data sets.

             Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)    66.190     6.224    10.635    80.586     0.000     0.289     0.243
read1           0.505     0.046    10.960    71.143     0.000     0.363     0.286
lrnprob1       -0.251     0.122    -2.056    77.657     0.043     0.310     0.256
behsymp1       -0.184     0.106    -1.727    82.939     0.088     0.273     0.233

                  Estimate
Residual~~Residual   90.074

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.

                 2.5 %       97.5 %
(Intercept) 53.8052534 78.575035473
read1        0.4130783  0.596798319
lrnprob1    -0.4944088 -0.007959306
behsymp1    -0.3955499  0.027924710
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals

who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (t = 11.73, p < .001). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets.
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     52.329       3 113.521   0.000   0.304

Hypothesis test adjusted for small samples with df=[134]
complete-data degrees of freedom.
```

13.6 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). The contents of this file were shown in Section 12.6.

The Mplus input file for analyzing the imputations is Ex13.inp.

Mplus Script Ex13.inp

```
1  DATA:
2  file = ./imps/implist.dat;
3  type = imputation;
4  VARIABLE:
5  names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
6    read1 read2 read3 read9 read9grp stanread7
7    math1 math2 math3 math9 math9grp stanmath7;
8  usevariables = read9 read1 lrnprob1 behsymp1;
9  MODEL:
```

```
read9 on read1 lrnprob1 behsymp1 (beta1-beta3);
MODEL TEST:
0 = beta1; 0 = beta2; 0 = beta3;
OUTPUT:
stdyx cinterval;
```

The script is virtually identical to the `Ex1.1.inp` file described in Section 1.2 with three exceptions. First, instead of naming the raw data set, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Finally, the `MODEL` section no longer specifies a normal distribution for the predictors. Readers can refer back to Section 1.2 for a detailed description of the other commands. The code block below shows the analysis and pooling script.

13.7 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus *F* test to evaluate the set of slope coefficients. The `MODEL TEST` command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and *p*-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       5

...

Wald Test of Parameter Constraints

          Value                            175.893
          Degrees of Freedom                     3
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
   READ1             0.506      0.043     11.868      0.000      0.182
   LRNPROB1         -0.231      0.113     -2.047      0.041      0.149
   BEHSYMP1         -0.189      0.101     -1.864      0.062      0.160

Intercepts
   READ9            65.487      5.803     11.284      0.000      0.150

Residual Variances
   READ9            86.366     11.202      7.710      0.000      0.138
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by .51 points on `READ9`. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 11.87$, $p < .001$). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Specifying the `stdyx` keyword with the `OPTIONS` command prints the table of standardized estimates and *R*-squared statistics shown below. The slope coefficients convey the expected change in standard deviation units for a one standard deviation increase in each predictor. For example, the model predicts that two individuals who differ by one standard deviation on `READ1` but are the same on `LRNPROB1` and `BEHSYMP1` should differ by 0.70 standard deviations on `READ9`. Collectively, the predictors explain 61% of the variation in ninth-grade reading scores.

```
STANDARDIZED MODEL RESULTS

STDYX Standardization

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
   READ1             0.701      0.044     15.767      0.000      0.102
   LRNPROB1         -0.168      0.082     -2.036      0.042      0.157
   BEHSYMP1         -0.153      0.082     -1.861      0.063      0.159

Intercepts
   READ9             4.424      0.531      8.332      0.000      0.152

Residual Variances
   READ9             0.394      0.055      7.166      0.000      0.099

R-SQUARE

   Observed                                         Two-Tailed   Rate of
   Variable       Estimate       S.E.  Est./S.E.    P-Value    Missing

   READ9             0.606      0.055     11.033      0.000      0.099
```

13.8 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex13.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex13.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
```

```
3     /imputation_ id male hispanic riskgrp atrisk
4       behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
5       math1 math2 math3 math9 math9grp stanmath7.
6   EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 7 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 8 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 9.

SPSS Script Ex13.spwb, continued

```
7   SORT CASES by imputation_.
8   SPLIT FILE layered by imputation_.
9   REGRESSION
10    /descriptives mean stddev corr sig n
11    /dependent read9
12    /method enter read1 lrnprob1 behsymp1.
```

13.9 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero (t = 10.96, p < .001). Note that these estimates are numerically equivalent to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	1	(Constant)	65.621	5.375		12.210	.000			
		read1	.494	.038	.701	12.934	.000			
		lrnprob1	-.222	.104	-.164	-2.131	.035			
		behsymp1	-.176	.093	-.146	-1.898	.060			
2.00	1	(Constant)	72.405	5.268		13.745	.000			
		read1	.468	.038	.671	12.223	.000			
		lrnprob1	-.323	.103	-.242	-3.146	.002			
		behsymp1	-.173	.091	-.146	-1.895	.060			
• • •										
20.00	1	(Constant)	59.772	5.837		10.239	.000			
		read1	.543	.041	.716	13.127	.000			
		lrnprob1	-.149	.112	-.101	-1.327	.187			
		behsymp1	-.223	.099	-.171	-2.245	.026			
Pooled	1	(Constant)	66.190	6.224		10.635	.000	.228	.289	.989
		read1	.505	.046		10.960	.000	.272	.363	.987
		lrnprob1	-.251	.122		-2.056	.041	.241	.310	.988
		behsymp1	-.184	.106		-1.727	.085	.218	.273	.989

a. Dependent Variable: read9

14

FCS Multiple Imputation: Regression with a Multicategorical Predictor

This example illustrates model-agnostic fully conditional specification multiple imputation with mixed variable types. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
RISKGRP	Emotional/behavioral disorder risk	2.2	1 = Low, 2 = Medium, 3 = High
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric

14.1 Imputation and Analysis Models

A common goal of model-agnostic imputation is to generate imputations for different purposes (e.g., descriptive summaries, several analyses within the same project). To illustrate multiple imputation with the fully conditional specification algorithm (i.e., multiple imputation by chained equations, or MICE; van Buuren, 2018), suppose that one use of the filled-in data sets involves a model where ninth grade broad reading scores are regressed on first grade reading achievement, teacher-rated learning problems and behavioral symptoms, and a three-category nominal variable indicating risk for emotional or behavioral disorders.

$$\begin{aligned} READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1) \\ + \beta_4(MEDRISK) + \beta_5(HIGHRISK) + \varepsilon \end{aligned} \tag{28}$$

The `MEDRISK` and `HIGHRISK` variables are dummy code variables that contrast the medium- and high-risk groups, respectively, against the low-risk reference group. Chapter 9 used the same analysis model to illustrate MCMC estimation and model-based multiple imputation.

Fully conditional specification uses a sequence of regression models to fill in missing values. Specifically, each MCMC iteration fits a series of models where one incomplete variable is regressed on all other variables. The predicted values and residual variance from each model define the center and spread of the imputed values, which are drawn at random from a normal distribution. After imputing the missing scores, the filled-in variable becomes a predictor in all other imputation models in the sequence. The imputation stage should include all variables and effects for the subsequent analyses, and it should incorporate auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). The imputation models use the five analysis variables and three auxiliary variables. Blimp uses the latent response variable framework (probit regression) for categorical variables like risk group (Enders et al., 2020).

14.2 Blimp and rblimp FCS Scripts

The code block below shows Blimp script `Ex14.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex14.imp

```
1   DATA: behaviorachievement.dat;
2   VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
3     read1 read2 read3 read9 read9grp stanread7
4     math1 math2 math3 math9 math9grp stanmath7;
5   NOMINAL: riskgrp;
6   MISSING: 999;
7   FCS: read9 read1 lrnprob1 behsymp1 riskgrp stanread7 read2;
8   SEED: 90291;
9   BURN: 1000;
10  ITERATIONS: 10000;
11  NIMPS: 20;
12  CHAINS: 20;
13  SAVE:
14  stacked = ./imps/imps.dat;
15  separate = ./imps/imp*.dat;
```

The first six lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The NOMINAL command on line 5 identifies the multicategorical nominal predictor, and the MISSING command on line 6 defines a global missing value code as 999.

Next, the FCS command lists all variables—complete or incomplete—that are included in the imputation phase. Using the FIXED command to identify complete variables reduces computational time because these variables do not require a regression model. Lines 8 through 10 can also be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the imputation model summaries are based (essentially, the total number of MCMC cycles across all chains following the warm-up period).

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed

data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

The corresponding `rblimp` script `Ex14.R` is shown below.

rblimp Script Ex14.R

```
library(rblimp)
load('behaviorachievement.rda')

mymodel <- rblimp_fcs(
  data = behaviorachievement,
  nominal = 'riskgrp',
  variables = 'read9 read1 lrnprob1 behsymp1 riskgrp stanread7 read2',
  seed = 90291,
  burn = 2000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp_fcs` function. The two exceptions are the `MISSING` and `FCS` commands. The former is omitted because that information is contained in the R data file. The `FCS` command is replaced by a `variables` parameter that lists the variables to be included in the imputation model. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like variable lists are enclosed in quotes, and numeric inputs like the seed and number of iterations do not require quotes. Finally, the `output(mymodel)` function prints the Blimp output.

14.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           51 to 100             1.417            45
                          101 to 200             1.156            45
                          151 to 300             1.243            45
                                 ...               ...            ..
                          901 to 1800            1.022            54
                          951 to 1900            1.021            45
                         1001 to 2000            1.021            45
```

The next output excerpt shows information about the data and the variables in the imputation models.

```
DATA INFORMATION:

  Sample Size:              138
  Missing Data Rates:

                 riskgrp = 02.17
                behsymp1 = 03.62
                lrnprob1 = 02.17
```

```
                    read1 = 06.52
                    read2 = 09.42
                    read9 = 17.39
                stanread7 = 19.57

Nominal Dummy Codes:

                  riskgrp = riskgrp.2 riskgrp.3


VARIABLES IN IMPUTATION MODEL:

  Incomplete continuous:  behsymp1 lrnprob1 read1 read2 read9 stanread7
  Incomplete nominal:     riskgrp

  NUMBER OF PARAMETERS
 Imputation Models:      68
```

MCMC estimation produces a distribution for each parameter in every unique imputation model. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. The Blimp output includes tables of regression parameters for every incomplete variable's imputation model. The imputation model parameters are not of substantive interest and would not be reported. An example table is shown below.

```
Missing variable:   behsymp1

Parameters                      Median     StdDev       2.5%      97.5%        PSR      N_Eff
-------------------------------------------------------------------------------------------

Grand Mean                      49.596      1.101     47.401     51.765      1.007   2184.881

Level 1:
  riskgrp.2                     -0.581      1.093     -2.657      1.578      1.006   3551.667
  riskgrp.3                      1.574      1.268     -1.031      3.876      1.009   1864.907
  lrnprob1                       0.705      0.079      0.547      0.856      1.005   4786.167
```

```
read1                          -0.217      0.093     -0.393     -0.032      1.004   3349.964
read2                           0.593      0.108      0.381      0.807      1.004   5270.399
read9                          -0.447      0.105     -0.652     -0.242      1.004   5961.720
stanread7                      -0.016      0.016     -0.049      0.016      1.003   5148.240
Residual Var.                  52.691      7.826     39.338     70.152      1.005   4634.464

                                ----------------------------------------------------------------
```

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

separate = './imps/imp*.dat'

   id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
   read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7

stacked = './imps/imps.dat'

    imp# id male hispanic riskgrp atrisk behsymp1 lrnprob1 read1 read2 read3
    read9 read9grp stanread7 math1 math2 math3 math9 math9grp stanmath7
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

14.4 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex14.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex14.R

```
1   library(rblimp)
2   library(mitml)
3   load('behaviorachievement.rda')
4
5   mymodel <- rblimp_fcs(...)
6
7   implist <- as.mitml(mymodel)
8
9   fit <- with(implist,
10    lm(read9 ~ read1 + lrnprob1 + behsymp1 + factor(riskgrp)))
11  estimates <- testEstimates(fit, extra.pars = T, df.com = 132)
12  estimates
13  confint(estimates)
14
15  null <- with(implist, lm(read9 ~ 1))
16  testModels(fit, null, df.com = 132, method = 'D1')
```

To begin, `as.mitml` on Line 7 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 9 and 10 fit the focal regression model using the `lm` function, and line 11 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. The `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data (i.e., $N–K–1$ degrees of freedom, where K is the number of predictors). This argument produces Barnard and Rubin degrees of freedom values. Lines 12 and 13 print the pooled estimates and confidence intervals. Finally, lines 15 and 16 specify a multiple imputation Wald F statistic evaluating the null hypothesis that all population slopes equal zero (Li et al., 1991). The test requires an additional model on line 15 that represents the null hypothesis, which in this case is an empty regression model with just an intercept. On line 16, the full model and null model objects passed into the `testModels` function, and the `D1` keyword requests the Wald test. As before, the `df.com` parameter is the denominator degrees of freedom that would have resulted had there been no missing data. This argument produces the Barnard and Rubin (1999) degrees of freedom adjustment.

14.5 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The RIV column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the FMI column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                 Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)        68.825     6.421    10.719    93.741     0.000     0.199     0.183
read1               0.481     0.047    10.287   117.185     0.000     0.076     0.086
lrnprob1           -0.248     0.117    -2.130    99.230     0.036     0.168     0.161
behsymp1           -0.169     0.100    -1.693   116.399     0.093     0.080     0.089
factor(riskgrp)2   -1.701     1.943    -0.876   115.037     0.383     0.087     0.095
factor(riskgrp)3   -2.677     2.700    -0.992    97.107     0.324     0.180     0.169

                  Estimate
Residual~~Residual   90.306

Hypothesis test adjusted for small samples with df=[132]
complete-data degrees of freedom.

                       2.5 %      97.5 %
(Intercept)      56.0763544 81.57453344
read1             0.3881339  0.57321747
lrnprob1         -0.4792725 -0.01693990
behsymp1         -0.3661746  0.02864639
factor(riskgrp)2 -5.5486638  2.14715922
factor(riskgrp)3 -8.0350660  2.68060322
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48

points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($t = 9.93$, $p < .001$). The two dummy codes appear as RISKGRP2 and RISKGRP3. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 3.12 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation.

Finally, the Wald omnibus F statistic is shown in the output table below. The test statistic is statistically significant, thus refuting the null hypothesis that all population slopes equal zero.

```
Model comparison calculated from 20 imputed data sets.
Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     34.144       5 125.765   0.000   0.157

Hypothesis test adjusted for small samples with df=[132]
complete-data degrees of freedom.
```

14.6 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). The contents of this file were shown in Section 12.6. The Mplus input file for analyzing the imputations is Ex14.inp.

Mplus Script Ex14.inp

```
1  DATA:
2  file = ./imps/implist.dat;
3  type = imputation;
4  VARIABLE:
5  names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
```

```
6     read1 read2 read3 read9 read9grp stanread7
7     math1 math2 math3 math9 math9grp stanmath7;
8   usevariables = read9 read1 lrnprob1 behsymp1 riskgrp2 riskgrp3;
9   DEFINE:
10  riskgrp2 = 0;
11  riskgrp3 = 0;
12  if(riskgrp eq 2) then riskgrp2 = 1;
13  if(riskgrp eq 3) then riskgrp3 = 1;
14  MODEL:
15  read9 on read1 lrnprob1 behsymp1 riskgrp2 riskgrp3 (beta1-beta5);
16  MODEL TEST:
17  0 = beta1; 0 = beta2; 0 = beta3;
18  OUTPUT:
19  stdyx cinterval;
```

The script is like previous Mplus scripts (e.g., the `Ex1.1.inp` file described in Section 1.2) with four exceptions. First, instead of naming the raw data set, the `DATA` command lists the text file containing the names of the imputed data sets (the `implist.dat` file located in the `./imps` subdirectory). The `type = imputation` subcommand instructs Mplus that the input data is a list of file names. Second, the missing subcommand is omitted because the analysis variables are now complete. Third, the `MODEL` section no longer specifies a normal distribution for the predictors or models for the auxiliary variables. Finally, lines 9 through 13 use the `DEFINE` command to create a pair of dummy codes. Lines 10 and 11 initialize a pair of new variables (`RISKGRP2` and `RISKGRP3`) with all 0s, and lines 12 and 13 recode these variables into dummy variables. Importantly, new variables computed with the `DEFINE` command must appear at the end of the `usevariables` list on line 8. The code block below shows the analysis and pooling script.

14.7 Mplus Output

When fitting regression models to complete data sets, researchers often use an omnibus F test to evaluate the set of slope coefficients. The `MODEL TEST` command specified a multiple imputation Wald chi-square statistic evaluating the null hypothesis that the population slopes equal zero (Asparouhov & Muthén, 2010b). The chi-square statistic, degrees of freedom, and p-value appear near the bottom of the `MODEL FIT INFORMATION` section under the `Wald Test of Parameter Constraints` heading. The test statistic is statistically significant, thus refuting the null hypothesis.

```
MODEL FIT INFORMATION

Number of Free Parameters                       7

...

Wald Test of Parameter Constraints

          Value                            173.432
          Degrees of Freedom                     5
          P-Value                           0.0000
```

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

READ9    ON
    READ1            0.481      0.046     10.501      0.000      0.074
    LRNPROB1        -0.248      0.114     -2.170      0.030      0.152
    BEHSYMP1        -0.169      0.098     -1.728      0.084      0.078
    RISKGRP2        -1.701      1.903     -0.894      0.372      0.084
    RISKGRP3        -2.677      2.649     -1.011      0.312      0.161

 Intercepts
    READ9           68.826      6.303     10.919      0.000      0.175

 Residual Variances
    READ9           86.381     12.116      7.129      0.000      0.265
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 10.50$, $p < .001$). The two dummy codes appear as RISKGRP2 and RISKGRP3. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 2.67 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC estimation. The output also includes a table with standardized coefficients and the *R*-squared statistic.

14.8 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the stacked keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is Ex14.spwb. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex14.spwb

```
CD '/users/username/desktop'.
DATA LIST free file =  'imps.dat'
 /imputation_ id male hispanic riskgrp atrisk
  behsymp1 lrnprob1 read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7.
EXE.

COMPUTE riskgrp2 = 0.
COMPUTE riskgrp3 = 0.
IF (riskgrp = 2) riskgrp2 = 1.
IF (riskgrp = 3) riskgrp3 = 1.
EXE.
```

The first line uses the CD command to change the working directory to the desktop. The username portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. The dummy codes for the RISKGRP variable are created beginning at line 8.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 13 sorts the data by the imputation index variable, and the SPLIT FILE command on line 14 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 15.

SPSS Script Ex14.spwb, continued

```
13  SORT CASES by imputation_.
14  SPLIT FILE layered by imputation_.
15  regression
16    /descriptives mean stddev corr sig n
17    /dependent read9
18    /method enter read1 lrnprob1 behsymp1 riskgrp2 riskgrp3.
```

14.9 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The regression output also includes pooled means and correlations. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on all other predictors should differ by 0.48 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically

different from zero (t = 10.29, p < .001). The two dummy codes appear as `RISKGRP2` and `RISKGRP3`. Consistent with a complete-data regression analysis, the dummy code slopes represent mean differences relative to the low-risk reference group. For example, holding all other predictors constant, the model predicts that a high-risk study would score 2.67 points lower than a low-risk student in the comparison group. Note that these estimates are virtually identical to those from MCMC and maximum likelihood estimation.

Coefficients[a]

imputation_	Model		Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	t	Sig.	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	1	(Constant)	69.856	5.812		12.019	.000			
		read1	.481	.045	.658	10.570	.000			
		lrnprob1	−.274	.106	−.200	−2.576	.011			
		behsymp1	−.155	.096	−.126	−1.604	.111			
		riskgrp2	−1.631	1.875	−.054	−.870	.386			
		riskgrp3	−3.476	2.497	−.099	−1.392	.166			
2.00	1	(Constant)	66.203	6.026		10.986	.000			
		read1	.485	.046	.671	10.599	.000			
		lrnprob1	−.182	.110	−.132	−1.646	.102			
		behsymp1	−.177	.098	−.144	−1.794	.075			
		riskgrp2	−2.382	1.905	−.080	−1.250	.213			
		riskgrp3	−3.062	2.545	−.087	−1.204	.231			
• • •										
20.00	1	(Constant)	67.727	5.919		11.442	.000			
		read1	.474	.045	.681	10.522	.000			
		lrnprob1	−.199	.111	−.147	−1.790	.076			
		behsymp1	−.184	.099	−.154	−1.868	.064			
		riskgrp2	−1.833	1.881	−.063	−.974	.332			
		riskgrp3	−1.399	2.530	−.041	−.553	.581			
Pooled	1	(Constant)	68.825	6.421		10.719	.000	.168	.199	.992
		read1	.481	.047		10.287	.000	.071	.076	.996
		lrnprob1	−.248	.117		−2.130	.033	.146	.168	.993
		behsymp1	−.169	.100		−1.693	.091	.074	.080	.996
		riskgrp2	−1.701	1.943		−.876	.381	.080	.087	.996
		riskgrp3	−2.677	2.700		−.992	.322	.155	.180	.992

a. Dependent Variable: read9

15

FCS Multiple Imputation: Random Intercept Model

This example illustrates model-agnostic fully conditional specification multiple imputation for multilevel data with random intercepts. The analysis uses the `problemsolving2level.dat` data set taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 928 student-level records include pretest and posttest math problem-solving and self-efficacy scores, standardized math scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
SCHOOL	School identifier	0	Integer index
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental
HISPANIC	Ethnicity/race	9.0	0 = Other, 1 = Hispanic
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Free/Reduced Lunch
PSOLVEPRE	Math problem-solving pretest	0	Numeric (37 to 66)
PSOLVEPST	Math problem-solving posttest	20.5	Numeric (37 to 65)

15.1 Imputation and Analysis Models

To illustrate multilevel fully conditional specification, suppose the ultimate analysis is a random intercept regression model. The goal of the analysis is to determine whether the intervention groups differ on an end-of-year math problem-solving test after controlling for three student-level covariates: math problem-solving pre-test scores, a Hispanic dummy code, and a free or

reduced lunch assistance dummy code. To convey each variable's level, the *i* and *j* subscripts denote students and schools, respectively.

$$\begin{aligned} PSOLVEPST_{ij} = &\left(\gamma_{00} + u_{0j}\right) + \gamma_{10}\left(PSOLVEPRE_{ij}^{cwc}\right) + \gamma_{20}\left(HISPANIC_{ij}^{cwc}\right) \\ &+ \gamma_{30}\left(FRLUNCH_{ij}^{cwc}\right) + \gamma_{01}\left(\mu_{j(PSOLVEPRE)}\right) + \gamma_{02}\left(\mu_{j(HISPANIC)}\right) \\ &+ \gamma_{03}\left(\mu_{j(FRLUNCH)}\right) + \gamma_{04}\left(CONDITION_{j}\right) + \varepsilon_{ij} \end{aligned} \tag{29}$$

The analysis model partitions the level-1 covariates into pure within-cluster (group mean centered) and between-cluster components. The *cwc* superscript denotes centering within cluster (group mean centering). All coefficients with a leading zero subscript are school-level effects, and all coefficients with non-zero leading subscripts are pure within-school effects. The γ_{04} slope is of particular interest because it captures the intervention effect, controlling for covariates. Chapter 16 uses the same analysis model to illustrate MCMC estimation and model-based multiple imputation.

Fully conditional specification imputation uses a sequence of univariate regression models to fill in missing values. Specifically, each MCMC iteration fits a series of models where one incomplete variable is regressed on all other variables. The predicted values and residual variance from each model define the center and spread of the imputed values, which are drawn at random from a normal distribution. After imputing the missing scores, the filled-in variable becomes a predictor in all other imputation models in the sequence. The imputation stage should include all variables and effects for the subsequent analyses, and it should incorporate auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Blimp's multilevel fully conditional specification routine also uses the latent cluster means (i.e., random intercepts) of level-1 variables in the imputation models (Enders et al., 2020). This disaggregated specification preserves unique within- and between-cluster associations in the data.

15.2 Blimp and rblimp FCS Scripts

The code block below shows Blimp script `Ex15.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex15.imp

```
1  DATA: problemsolving2level.dat;
2  VARIABLES: school student condition teachexp eslpct ethnic male
```

```
3     frlunch lowach stanmath efficacypre efficacypst psolvepre psolvepst;
4   CLUSTERID: school;
5   ORDINAL: condition frlunch;
6   MISSING: 999;
7   FIXED: condition psolvepre;
8   FCS: psolvepst psolvepre hispanic frlunch condition;
9   SEED: 90291;
10  BURN: 1000;
11  ITERATIONS: 10000;
12  NIMPS: 20;
13  CHAINS: 20;
14  SAVE:
15  stacked = ./imps/imps.dat;
16  separate = ./imps/imp*.dat;
```

The first six lines can be viewed as a set of commands that specify information about the data and variables. The `DATA` command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the `VARIABLES` command names the data columns. The `CLUSTERID` command on line 4 lists the school-level identifier variable that indicates the clustering of the data records in schools. Including the `CLUSTERID` command automatically introduces random intercepts for all level-1 variables. When a level-1 variable appears as a predictor of another level-1 variable, its random intercepts are used as a level-2 covariate in the imputation model (i.e., imputation uses latent contextual effects). When a level-1 variable appears as a predictor of a level-2 variable, just the random intercepts are in the imputation model. The `ORDINAL` command on line 5 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The `MISSING` command on line 6 defines a global missing value code as 999.

Next, the `FCS` command lists all variables—complete or incomplete at either level—that are included in the imputation phase. Using the `FIXED` command to identify complete variables reduces computational time because these variables do not require a regression model. Lines 9 through 11 can also be viewed as a block of commands that specify features of the MCMC algorithm: the `SEED` command gives an integer string that initializes the random number generator, the `BURN` command specifies the number of iterations for the warm-up or burn-in period, and the `ITERATIONS` command gives the number of MCMC iterations on which the

imputation model summaries are based (essentially, the total number of MCMC cycles across all chains following the warm-up period).

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

The corresponding rblimp script Ex15.R is shown below.

rblimp Script Ex15.R

```
library(rblimp)
load('problemsolving2level.rda')

mymodel <- rblimp_fcs(
  data = problemsolving2level,
  clusterid = 'school',
  ordinal = 'condition hispanic frlunch',
  fixed = 'condition psolvepre',
  variables = 'psolvepst psolvepre hispanic frlunch condition',
  seed = 90291,
  burn = 1000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp_fcs function. The two exceptions are the MISSING and FCS commands. The former is omitted because that information is contained in the R data file. The FCS command is replaced by a variables parameter that lists the variables to be included in the imputation model. Following

R convention, the input parameters are separated by commas. Alphanumeric inputs like variable lists are enclosed in quotes, and numeric inputs like the seed and number of iterations do not require quotes. Finally, the `output(mymodel)` function prints the Blimp output.

15.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           26 to 50              1.374            14
                           51 to 100             1.300            19
                           76 to 150             1.125            12
                                  ...              ...            ..
                          451 to 900             1.022            12
                          476 to 950             1.019            12
                          501 to 1000            1.015             1
```

The next output excerpt shows information about the data and the variables in the imputation models.

```
DATA INFORMATION:

  Level-2 identifier:       school
  Sample Size:              982
  Level-2 Clusters:         29
```

```
  Missing Data Rates:

                hispanic = 08.96
                 frlunch = 04.68
               psolvepst = 20.47

VARIABLES IN IMPUTATION MODEL:

  Fixed variables:        condition psolvepre
  Incomplete continuous:  psolvepst
  Incomplete ordinal:     hispanic frlunch

  NUMBER OF PARAMETERS
    Imputation Models:      25
```

MCMC estimation produces a distribution for each parameter in every unique imputation model. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. The Blimp output includes tables of regression parameters for every incomplete variable's imputation model. The imputation model parameters are not of substantive interest and would not be reported. An example table is shown below.

```
  Missing variable:   frlunch

  Parameters                      Median     StdDev       2.5%      97.5%        PSR      N_Eff
                                ------------------------------------------------------------------

  Grand Mean                       1.013      0.138      0.754      1.295      1.008   2339.218

  Level 1:
    hispanic                       0.112      0.067     -0.015      0.248      1.018   1013.866
    psolvepre                      0.010      0.012     -0.013      0.034      1.009   1991.112
    psolvepst                     -0.025      0.013     -0.050     -0.000      1.012   1781.354
    Residual Var.                  1.000      0.000      1.000      1.000        nan        nan

  Level 2:
    condition                      0.093      0.360     -0.620      0.822      1.005   3010.372
    hispanic                       0.016      0.256     -0.473      0.538      1.007   2250.809
```

```
  psolvepst                      -0.124      0.102     -0.333      0.066      1.005   2603.149
  Residual Var.                   0.269      0.111      0.136      0.563      1.010   1860.140
Thresholds:
  Tau 1                           0.000      0.000      0.000      0.000        nan        nan

                                ------------------------------------------------------------
```

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     school student condition teachexp eslpct ethnic male frlunch
     lowach stanmath efficacyp efficacy1 psolvepre psolvepst

   stacked = './imps/imps.dat'

     imp# school student condition teachexp eslpct ethnic male frlunch
     lowach stanmath efficacyp efficacy1 psolvepre psolvepst
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

15.4 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex15.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex15.R

```
1   library(rblimp)
2   library(rockchalk)
3   library(lme4)
4   library(mitml)
5   load('problemsolving2level.rda')
6
7   mymodel <- rblimp(...)
8
9   implist <- as.mitml(mymodel)
10
11  for (i in 1:length(implist)) {
12    implist[[i]] <- gmc(implist[[i]], x = c('psolvepre','hispanic','frlunch'),
13      by = c('school'), FUN = mean, suffix = c('.meanj', '.cwc'),
14      fulldataframe = TRUE)
15  }
16
17  fit <- with(implist,
18    lmer(psolvepst ~ psolvepre.cwc + hispanic.cwc + frlunch.cwc
19    + psolvepre.meanj + hispanic.meanj + frlunch.meanj + condition
20    + (1 | school), REML = T))
21
22  estimates <- testEstimates(fit, extra.pars = T)
23  estimates
24  confint(estimates)
```

To begin, `as.mitml` on Line 9 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 11 through 15 use the `gmc` function in the `rockchalk` package to group mean center three predictors at their manifest (arithmetic) cluster means. Lines 17 through 20 fit the focal regression model using the `lmer` function, and line 22 uses the `testEstimates` function in `mitml` to implement Rubin's pooling rules and save the results in an object called `estimates`. Lines 23 and 24 print the pooled estimates and confidence intervals.

15.5 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth

columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)       20.215    11.486     1.760  5472.720     0.078     0.063     0.059
psolvepre.cwc      0.457     0.035    13.235   683.373     0.000     0.200     0.169
hispanic.cwc       1.024     0.416     2.462   424.087     0.014     0.268     0.215
frlunch.cwc       -0.714     0.492    -1.451   158.111     0.149     0.531     0.355
psolvepre.meanj    0.627     0.218     2.880  4712.621     0.004     0.068     0.064
hispanic.meanj     4.976     1.382     3.601  4473.786     0.000     0.070     0.066
frlunch.meanj     -2.644     2.506    -1.055  6610.732     0.292     0.057     0.054
condition          2.371     0.720     3.295  2326.583     0.001     0.099     0.091

                           Estimate
Intercept~~Intercept|school    2.303
Residual~~Residual            20.609
ICC|school                     0.101

Unadjusted hypothesis test as appropriate in larger samples.

> confint(estimates)
                     2.5 %     97.5 %
(Intercept)     -2.3018772 42.7319331
psolvepre.cwc    0.3891564  0.5247364
hispanic.cwc     0.2063039  1.8414960
frlunch.cwc     -1.6863210  0.2578444
psolvepre.meanj  0.2001714  1.0534617
hispanic.meanj   2.2667656  7.6850867
frlunch.meanj   -7.5570658  2.2693889
condition        0.9599550  3.7818573
```

The random intercept and within-cluster residual variances are denoted `Intercept~~Intercept|school` and `Residual~~Residual`, respectively. Moving to the coefficient section, the primary focus is the γ_{04} coefficient, which indicates that intervention

schools scored 2.37 points higher than control schools, on average, controlling for student- and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero ($t = 3.30$, $p < .001$).

15.6 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). The contents of this file were shown in Section 12.6. The Mplus input file for analyzing the imputations is `Ex15.inp`.

Mplus Script Ex15.inp

```
DATA:
file = ./imps/implist.dat;
type = imputation;
VARIABLE:
names = school student condition teachexp eslpct ethnic male frlunch
   lowach stanmath efficacy1 efficacy2 psolvepre psolvepst;
usevariables = psolvepst psolvepre hispanic frlunch condition
    psolveprej hispanicj frlunchj;
cluster = school;
within = psolvepre hispanic frlunch;
between = psolveprej hispanicj frlunchj condition;
DEFINE:
psolveprej = cluster_mean(psolvepre);
hispanicj = cluster_mean(hispanic);
frlunchj = cluster_mean(frlunch);
center psolvepre hispanic frlunch (groupmean);
center psolveprej hispanicj frlunchj (grandmean);
ANALYSIS:
type = twolevel;
MODEL:
%within%
psolvepst on psolvepre hispanic frlunch;
%between%
psolvepst on psolveprej hispanicj frlunchj condition;
```

```
24  OUTPUT:
25  stdyx cinterval;
```

The DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand instructs Mplus that the input data is a list of file names. The VARIABLE command provides information about the data. Beginning on line 5, the names subcommand assigns names to the variables in the input data file, and the usevariables subcommand selects variables for the analysis, with new variables computed on the DEFINE command listed at the end of the line. The cluster command on line 8 lists the school-level identifier variable that indicates the clustering of the data records in schools. The within and between subcommands on lines 9 and 10 identify level-1 and level-2 predictors, respectively. Lines 11 through 14 define new variables that are the group means of the level-1 covariates. On lines 15 and 16, the center subcommand under the DEFINE command centers the within- and between-cluster covariates at their group and grand means, respectively. The ANALYSIS command and the type = twolevel subcommand is required for estimating two-level models. The MODEL section of the script consists of two sections: the %within% section specifies the regression of the outcome on level-1 predictors, and the %between% section specifies the regression of the random intercepts on the level-2 predictors. Finally, the OUTPUT command specifies two keywords on line 25 that request standardized coefficients and confidence intervals.

15.7 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The Rate of Missing column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS
                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

Within Level
```

```
 PSOLVEPST  ON
    PSOLVEPRE          0.457      0.032     14.287      0.000      0.197
    HISPANIC           1.024      0.379      2.701      0.007      0.260
    FRLUNCH           -0.714      0.469     -1.524      0.128      0.392

 Residual Variances
    PSOLVEPST         20.539      1.393     14.747      0.000      0.212

Between Level

 PSOLVEPST  ON
    PSOLVEPREJ         0.629      0.159      3.953      0.000      0.119
    HISPANICJ          4.980      1.284      3.878      0.000      0.075
    FRLUNCHJ          -2.618      1.687     -1.552      0.121      0.118
    CONDITION          2.372      0.660      3.594      0.000      0.107

 Intercepts
    PSOLVEPST         52.516      0.520    101.066      0.000      0.072

 Residual Variances
    PSOLVEPST          1.816      0.746      2.436      0.015      0.100
```

Mplus separates the level-1 and level-2 effects on the output (labeled `Within Level` and `Between Level`, respectively). The primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.37 points higher than control schools, on average, controlling for student- and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero ($z = 3.59$, $p < .001$). Note that these estimates are virtually identical to those from MCMC estimation in the next section.

15.8 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex15.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex15.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3    /imputation_ school student condition teachexp eslpct hispanic male
4     frlunch lowach stanmath efficacypre efficacypst psolvepre psolvepst.
5  EXE.
6
7  AGGREGATE
8    /outfile = * mode = addvariables overwrite = yes
9    /break=imputation_ school
10   /psolvepre_meanj = mean(psolvepre)
11   /hispanic_meanj = mean(hispanic)
12   /frlunch_meanj = mean(frlunch).
13 EXE.
14
15 AGGREGATE
16   /outfile = * mode = addvariables overwrite = yes
17   /psolvepre_mean = mean(psolvepre)
18   /hispanic_mean = mean(hispanic)
19   /frlunch_mean = mean(frlunch).
20 EXE.
21
22 COMPUTE psolvepre_cwc = psolvepre - psolvepre_meanj.
23 COMPUTE hispanic_cwc = hispanic - hispanic_meanj.
24 COMPUTE frlunch_cwc = frlunch - frlunch_meanj.
25 COMPUTE psolvepre_meanj = psolvepre_meanj - psolvepre_mean.
26 COMPUTE hispanic_meanj = hispanic_meanj - hispanic_mean.
27 COMPUTE frlunch_meanj = frlunch_meanj - frlunch_mean.
28 EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. The `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named `IMPUTATION_` is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 7, the `AGGREGATE` command adds the grand means to the data, whereas the `AGGREGATE` command on

line 15 adds the grand means into the data. Beginning on line 22, level-1 predictors are centered at their group means, and level-2 predictors are centered at their grand means.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The `SORT` command on line 29 sorts the data by the imputation index variable, and the `SPLIT FILE` command on line 30 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 31.

SPSS Script Ex15.spwb, continued

```
29  SORT CASES by imputation_.
30  SPLIT FILE layered by imputation_.
31  MIXED psolvepst with psolvepre_cwc hispanic_cwc frlunch_cwc
32     psolvepre_meanj hispanic_meanj frlunch_meanj condition
33    /print = solution testcov
34    /fixed = intercept psolvepre_cwc hispanic_cwc frlunch_cwc
35     psolvepre_meanj hispanic_meanj frlunch_meanj condition
36    /random = intercept | subject(school) covtype(id).
```

15.9 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The output also includes pooled estimates of level-2 variance and covariance parameters. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data multilevel analysis. The pooled regression coefficients are in the table labeled `Estimates of Fixed Effects`. The primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.39 points higher than control schools, on average, controlling for student-and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero ($t = 3.36$, $p < .001$). The pooled random intercept and within-cluster residual variances are in the table labeled `Estimates of Covariance Parameters`.

Estimates of Fixed Effects[a]

imputation_	Parameter	Estimate	Std. Error	df	t	Sig.	95% Confidence Interval		Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
							Lower Bound	Upper Bound			
1.00	Intercept	52.580433	.495959	26.074	106.018	.000	51.561115	53.599750			
	psolvepre_cwc	.467157	.031442	951.185	14.858	.000	.405452	.528861			
	hispanic_cwc	1.338000	.364458	951.185	3.671	.000	.622766	2.053234			
	frlunch_cwc	-.645617	.396321	951.185	-1.629	.104	-1.423381	.132148			
	psolvepre_meanj	.563848	.210741	22.763	2.676	.014	.127645	1.000051			
	hispanic_meanj	5.138691	1.321169	26.070	3.890	.001	2.423344	7.854039			
	frlunch_meanj	-2.426586	2.278931	22.311	-1.065	.298	-7.148983	2.295810			
	condition	2.752637	.674008	24.410	4.084	.000	1.362789	4.142486			
2.00	Intercept	52.722033	.554059	25.758	95.156	.000	51.582627	53.861439			
	psolvepre_cwc	.438724	.031450	951.062	13.950	.000	.377006	.500443			
	hispanic_cwc	.729561	.367351	951.062	1.986	.047	.008649	1.450474			
	frlunch_cwc	-.841150	.403055	951.062	-2.087	.037	-1.632130	-.050169			
	psolvepre_meanj	.644775	.230154	23.088	2.801	.010	.168767	1.120784			
	hispanic_meanj	5.078729	1.452633	25.164	3.496	.002	2.087966	8.069493			
	frlunch_meanj	-2.810802	2.700134	22.358	-1.041	.309	-8.405346	2.783742			
	condition	2.537438	.753748	24.403	3.366	.003	.983136	4.091740			
• • •											
20.00	Intercept	52.668203	.507763	25.852	103.726	.000	51.624191	53.712215			
	psolvepre_cwc	.471265	.030876	951.009	15.263	.000	.410672	.531857			
	hispanic_cwc	.729314	.360138	951.009	2.025	.043	.022556	1.436072			
	frlunch_cwc	-.626613	.389548	951.009	-1.609	.108	-1.391087	.137860			
	psolvepre_meanj	.674558	.214997	22.827	3.138	.005	.229617	1.119500			
	hispanic_meanj	5.229732	1.308554	25.502	3.997	.000	2.537403	7.922061			
	frlunch_meanj	-1.521656	2.408127	22.697	-.632	.534	-6.506934	3.463622			
	condition	2.521599	.693632	24.259	3.635	.001	1.090821	3.952377			
Pooled	Intercept	52.712916	.519739		101.422	.000	51.694050	53.731782	.055	.058	.997
	psolvepre_cwc	.457342	.033487		13.657	.000	.391644	.523041	.126	.142	.994
	hispanic_cwc	.883800	.432438		2.044	.042	.031909	1.735691	.288	.394	.986
	frlunch_cwc	-.762919	.448334		-1.702	.090	-1.644290	.118452	.221	.278	.989
	psolvepre_meanj	.647754	.214709		3.017	.003	.226889	1.068619	.041	.042	.998
	hispanic_meanj	4.945729	1.369106		3.612	.000	2.261890	7.629569	.051	.053	.997
	frlunch_meanj	-2.183826	2.482914		-.880	.379	-7.051046	2.683394	.051	.053	.997
	condition	2.385462	.709734		3.361	.001	.994050	3.776875	.064	.068	.997

a. Dependent Variable: psolvepst.

Estimates of Covariance Parameters[a]

nputation_	Parameter		Estimate	Std. Error	Wald Z	Sig.	95% Confidence Interval		Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
							Lower Bound	Upper Bound			
00	Residual		20.502350	.940126	21.808	.000	18.740112	22.430301			
	Intercept [subject = school]	Variance	2.275789	.839452	2.711	.007	1.104475	4.689301			
00	Residual		20.552490	.942486	21.807	.000	18.785833	22.485286			
	Intercept [subject = school]	Variance	2.887200	1.016744	2.840	.005	1.447843	5.757478			
• •											
0.00	Residual		19.810977	.908508	21.806	.000	18.108014	21.674095			
	Intercept [subject = school]	Variance	2.427108	.879182	2.761	.006	1.193314	4.936550			
ooled	Residual		20.388138	1.059107		.000	18.305865	22.470410	.225	.283	.98
	Intercept [subject = school]	Variance	2.314333	.884311		.009	.580328	4.048339	.085	.092	.99

a. Dependent Variable: psolvepst.

16

MCMC: Random Intercept Model

This example illustrates a two-level multiple regression with random intercepts. The analysis uses the `problemsolving2level.dat` data set taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 928 student-level records include pretest and posttest math problem-solving and self-efficacy scores, standardized math scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
SCHOOL	School identifier	0	Integer index
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental
HISPANIC	Ethnicity/race	9.0	0 = Other, 1 = Hispanic
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Free/Reduced Lunch
PSOLVEPRE	Math problem-solving pretest	0	Numeric (37 to 66)
PSOLVEPST	Math problem-solving posttest	20.5	Numeric (37 to 65)

16.1 Analysis Model

The analysis is a random intercept model with a school-level intervention code and three student-level covariates: math problem-solving pre-test scores, a Hispanic dummy code, and a free or reduced lunch assistance dummy code. The goal of the analysis is to determine whether the intervention groups differ on an end-of-year math problem-solving test after controlling for three student-level variables. To convey each variable's level, the i and j subscripts denote students and schools, respectively.

$$\begin{aligned} PSOLVEPST_{ij} = (\gamma_{00} + u_{0j}) + \gamma_{10}\left(PSOLVEPRE_{ij}^{cwc}\right) + \gamma_{20}\left(HISPANIC_{ij}^{cwc}\right) \\ + \gamma_{30}\left(FRLUNCH_{ij}^{cwc}\right) + \gamma_{01}\left(\mu_{j(PSOLVEPRE)}^{cgm}\right) + \gamma_{02}\left(\mu_{j(HISPANIC)}^{cgm}\right) \\ + \gamma_{03}\left(\mu_{j(FRLUNCH)}^{cgm}\right) + \gamma_{04}\left(CONDITION_j\right) + \varepsilon_{ij} \end{aligned} \tag{30}$$

The analysis model partitions the level-1 covariates into pure within-cluster and between-cluster components. MCMC centers the student-level covariates at their school-level latent group means. All coefficients with a leading zero subscript are school-level effects, and all coefficients with non-zero leading subscripts are pure within-school effects. The *cwc* superscript denotes centering within cluster (group mean centering), and *cgm* indicates centering at the grand mean. The γ_{04} slope is of particular interest because it captures the intervention effect, controlling for covariates. Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Blimp uses a factored regression specification that assigns separate distributions to the predictors and outcome. By default, Blimp invokes a multivariate normal distribution for numeric predictors and the latent response scores for discrete predictors.

16.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex16.1.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex16.1.imp

```
DATA: problemsolving2level.dat;
VARIABLES: school student condition teachexp eslpct ethnic male
  frlunch lowach stanmath efficacypre efficacypst psolvepre psolvepst;
CLUSTERID: school;
ORDINAL: condition hispanic frlunch;
MISSING: 999;
FIXED: condition psolvepre;
CENTER:
groupmean = psolvepre hispanic frlunch;
grandmean = psolvepre.mean hispanic.mean frlunch.mean;
MODEL:
psolvepst ~ psolvepre hispanic frlunch
   psolvepre.mean hispanic.mean frlunch.mean condition;
```

```
14  SEED: 90291;
15  BURN: 5000;
16  ITERATIONS: 10000;
```

The first six lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The CLUSTERID command on line 4 lists the school-level identifier variable that indicates the clustering of the data records in schools. Including the CLUSTERID command automatically introduces random intercepts. The ORDINAL command on line 5 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The MISSING command on line 6 defines a global missing value code as 999.

The FIXED, CENTER, and MODEL blocks can be viewed as a set. The FIXED command identifies two complete predictors that do not require a distribution or regression model. The CENTER command deviates the student-level covariates at the latent group means, and it centers the group means (ending in the .mean suffix) at their iteratively-estimated grand means. Beginning on line 11, the MODEL command lists the regression models, with outcome variables to the left of the tilde and predictors to the right. Blimp automatically configures the explanatory variable models under the assumption that the numeric variables and latent response scores (discrete predictors) are normally distributed. Finally, lines 14 through 16 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex16.R is shown below.

rblimp Script Ex16.R

```
1  library(rblimp)
2  load('problemsolving2level.rda')
3
4  mymodel <- rblimp(
5    data = problemsolving2level,
```

```
    clusterid = 'school',
    ordinal = 'condition hispanic frlunch',
    fixed = 'condition psolvepre',
    center = 'groupmean = psolvepre hispanic frlunch;
      grandmean = psolvepre.mean hispanic.mean frlunch.mean',
    model = 'psolvepst ~ psolvepre hispanic frlunch
      psolvepre.mean hispanic.mean frlunch.mean condition',
    seed = 90291,
    burn = 5000,
    iter = 10000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp` function. The two exceptions are the `VARIABLES` and `MISSING` commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the `MODEL` command) are separated by semicolons, as they are in the Blimp script. Finally, the `output(mymodel)` function prints the Blimp output.

16.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.
```

```
   Comparing iterations across 2 chains     Highest PSR   Parameter #
                          126 to 250             1.340            21
                          251 to 500             1.252            39
                          376 to 750             1.090            31
                                 ...               ...            ..
                         2251 to 4500            1.017             3
                         2376 to 4750            1.027            21
                         2501 to 5000            1.029            21
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3, but it additionally reports the number of observations at each level.

```
DATA INFORMATION:

  Level-2 identifier:       school
  Sample Size:              982
  Level-2 Clusters:         29
  Missing Data Rates:

               psolvepst = 20.47
                hispanic = 08.96
                 frlunch = 04.68
               psolvepre = 00.00
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as

"computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  psolvepst

Grand Mean Centered: frlunch.mean[school] hispanic.mean[school]
                     psolvepre.mean[school]
Group Mean Centered: frlunch hispanic psolvepre
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
L2 : Var(Intercept)	2.452	1.204	1.056	5.679	---	---	1460.938
Residual Var.	20.586	1.073	18.622	22.787	---	---	5882.801
Coefficients:							
Intercept	52.423	0.732	50.928	53.855	5125.493	0.000	335.563
psolvepre	0.456	0.036	0.387	0.527	160.387	0.000	5660.178
hispanic	0.944	0.437	0.087	1.802	4.672	0.031	4201.346
frlunch	-0.782	0.469	-1.698	0.144	2.790	0.095	4545.674
psolvepre.mean[school]	0.710	0.335	0.062	1.405	4.585	0.032	494.808
hispanic.mean[school]	5.312	1.706	2.014	8.678	9.694	0.002	756.422
frlunch.mean[school]	-1.698	3.519	-8.737	5.161	0.234	0.629	553.383
condition	2.423	0.791	0.900	4.017	9.415	0.002	876.374
Standardized Coefficients:							
psolvepre	0.361	0.028	0.306	0.414	169.280	0.000	3072.827
hispanic	0.064	0.030	0.006	0.122	4.673	0.031	4124.638
frlunch	-0.049	0.029	-0.106	0.009	2.784	0.095	4432.903
psolvepre.mean[school]	0.195	0.087	0.017	0.364	4.982	0.026	523.214
hispanic.mean[school]	0.225	0.069	0.085	0.355	10.536	0.001	796.874
frlunch.mean[school]	-0.043	0.085	-0.207	0.125	0.245	0.620	553.606
condition	0.201	0.063	0.075	0.321	10.150	0.001	885.531

```
Proportion Variance Explained
```

```
by Coefficients                0.332     0.040     0.253     0.409       ---       ---  1034.513
by Level-2 Random Intercepts   0.071     0.031     0.031     0.149       ---       ---  1432.122
by Level-1 Residual Variation  0.594     0.041     0.505     0.669       ---       ---  1142.265

                               ------------------------------------------------------------------------------
```

To begin, the `N_Eff` values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the `N_Eff` values are insufficient, increasing the value on the `ITERATIONS` command will remedy the issue.

The results are interpreted in the same way as a complete-data multilevel analysis. The first section of the output table displays the variance estimates. The random intercept and within-cluster residual variances are denoted `L2:Var(Intercept)` and `Residual Var.`, respectively. Moving to the coefficient section, the primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.42 points higher than control schools, on average, controlling for student- and school-level covariates. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. Finally, the bottom section of the table displays Rights and Sterba (2019) R-squared effect size values. The fixed effects explain 33% of the total variation, and the random intercepts account for 7% of the variability.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

16.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex16.2.imp` is identical `Ex16.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The NIMPS, CHAINS and SAVE commands can be viewed as a set. Setting NIMPS equal to CHAINS saves a single filled-in data set from the final iteration of a unique MCMC process, thus avoiding autocorrelation among the imputations. The SAVE command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the ./imps part of the file path points to a subfolder named imps located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     school student condition teachexp eslpct ethnic male frlunch
     lowach stanmath efficacyp efficacy1 psolvepre psolvepst

   stacked = './imps/imps.dat'

     imp# school student condition teachexp eslpct ethnic male frlunch
     lowach stanmath efficacyp efficacy1 psolvepre psolvepst
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex16.R

```
library(rblimp)
load('problemsolving2level.rda')

mymodel <- rblimp(
  data = problemsolving2level,
  clusterid = 'school',
  ordinal = 'condition hispanic frlunch',
  fixed = 'condition psolvepre',
  center = 'groupmean = psolvepre hispanic frlunch;
    grandmean = psolvepre.mean hispanic.mean frlunch.mean',
  model = 'psolvepst ~ psolvepre hispanic frlunch
    psolvepre.mean hispanic.mean frlunch.mean condition',
  seed = 90291,
  burn = 5000,
  iter = 10000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

16.5 Analyzing Multiple Imputations in R

Continuing with the previous rblimp script, the following excerpt from Ex7.R shows how to perform multiple imputation inference. The script requires the mitml package (Grund et al., 2023).

R Script Ex16.R

```
1   library(rblimp)
2   library(rockchalk)
3   library(lme4)
4   library(mitml)
5   load('problemsolving2level.rda')
6
7   mymodel <- rblimp(...)
8
9   implist <- as.mitml(mymodel)
10
11  for (i in 1:length(implist)) {
12    implist[[i]] <- gmc(implist[[i]], x = c('psolvepre','hispanic','frlunch'),
13      by = c('school'), FUN = mean, suffix = c('.meanj', '.cwc'),
14      fulldataframe = TRUE)
15  }
16
17  fit <- with(implist,
18    lmer(psolvepst ~ psolvepre.cwc + hispanic.cwc + frlunch.cwc
19    + psolvepre.meanj + hispanic.meanj + frlunch.meanj + condition
20    + (1 | school), REML = T))
21
22  estimates <- testEstimates(fit, extra.pars = T)
23  estimates
24  confint(estimates)
```

To begin, as.mitml on Line 9 is an rblimp function that converts the imputation object into a list of data sets called implist, as required by the mitml package. Lines 11 through 15 use the gmc function in the rockchalk package to group mean center three predictors at their manifest (arithmetic) cluster means. Lines 17 through 20 fit the focal regression model using the lmer function, and line 22 uses the testEstimates function in mitml to implement Rubin's pooling

rules and save the results in an object called `estimates`. Lines 23 and 24 print the pooled estimates and confidence intervals.

16.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)       18.844    11.351     1.660 11720.210     0.097     0.042     0.040
psolvepre.cwc      0.457     0.033    13.657  1229.129     0.000     0.142     0.126
hispanic.cwc       0.884     0.432     2.044   238.166     0.042     0.394     0.288
frlunch.cwc       -0.763     0.448    -1.702   402.169     0.090     0.278     0.221
psolvepre.meanj    0.648     0.215     3.017 11590.269     0.003     0.042     0.041
hispanic.meanj     4.946     1.369     3.612  7380.527     0.000     0.053     0.051
frlunch.meanj     -2.184     2.483    -0.880  7373.485     0.379     0.053     0.051
condition          2.385     0.710     3.361  4675.153     0.001     0.068     0.064

                          Estimate
Intercept~~Intercept|school    2.314
Residual~~Residual            20.388
ICC|school                     0.102

Unadjusted hypothesis test as appropriate in larger samples.

                    2.5 %     97.5 %
(Intercept)     -3.4048602 41.0932538
psolvepre.cwc    0.3916436  0.5230409
hispanic.cwc     0.0319088  1.7356905
frlunch.cwc     -1.6442904  0.1184517
psolvepre.meanj  0.2268885  1.0686187
hispanic.meanj   2.2618899  7.6295689
frlunch.meanj   -7.0510463  2.6833948
```

```
condition        0.9940498  3.7768751
```

The random intercept and within-cluster residual variances are denoted `Intercept~~Intercept|school` and `Residual~~Residual`, respectively. Moving to the coefficient section, the primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.39 points higher than control schools, on average, controlling for student- and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero (t = 3.36, p < .001). Note that the intercept differs prior the MCMC estimate because the cluster means are not centered in the multiple imputation analysis.

16.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `separate` keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` subfolder). The contents of this file were shown in Section 12.6. The Mplus input file for analyzing the imputations is `Ex16.inp`.

Mplus Script Ex16.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = school student condition teachexp eslpct ethnic male frlunch
6     lowach stanmath efficacy1 efficacy2 psolvepre psolvepst;
7   usevariables = psolvepst psolvepre hispanic frlunch condition
7      psolveprej hispanicj frlunchj;
8   cluster = school;
9   within = psolvepre hispanic frlunch;
10  between = psolveprej hispanicj frlunchj condition;
11  DEFINE:
12  psolveprej = cluster_mean(psolvepre);
13  hispanicj = cluster_mean(hispanic);
14  frlunchj = cluster_mean(frlunch);
```

```
15  center psolvepre hispanic frlunch (groupmean);
16  center psolveprej hispanicj frlunchj (grandmean);
17  ANALYSIS:
18  type = twolevel;
19  MODEL:
20  %within%
21  psolvepst on psolvepre hispanic frlunch;
22  %between%
23  psolvepst on psolveprej hispanicj frlunchj condition;
24  OUTPUT:
25  stdyx cinterval;
```

The DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand instructs Mplus that the input data is a list of file names. The VARIABLE command provides information about the data. Beginning on line 5, the names subcommand assigns names to the variables in the input data file, and the usevariables subcommand selects variables for the analysis, with new variables computed on the DEFINE command listed at the end of the line. The cluster command on line 8 lists the school-level identifier variable that indicates the clustering of the data records in schools. The within and between subcommands on lines 9 and 10 identify level-1 and level-2 predictors, respectively. Lines 11 through 14 define new variables that are the group means of the level-1 covariates. On lines 15 and 16, the center subcommand under the DEFINE command centers the within- and between-cluster covariates at their group and grand means, respectively. The ANALYSIS command and the type = twolevel subcommand is required for estimating two-level models. The MODEL section of the script consists of two sections: the %within% section specifies the regression of the outcome on level-1 predictors, and the %between% section specifies the regression of the random intercepts on the level-2 predictors. Finally, the OUTPUT command specifies two keywords on line 25 that request standardized coefficients and confidence intervals.

16.8 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding z-statistics and p-values. The focal model results are shown below. The Rate of Missing column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

Within Level

 PSOLVEPST  ON
    PSOLVEPRE        0.457      0.030     15.065      0.000      0.153
    HISPANIC         0.884      0.397      2.227      0.026      0.343
    FRLUNCH         -0.763      0.407     -1.875      0.061      0.269

 Residual Variances
    PSOLVEPST       20.319      1.309     15.523      0.000      0.145

Between Level

 PSOLVEPST  ON
    PSOLVEPREJ       0.650      0.153      4.240      0.000      0.079
    HISPANICJ        4.949      1.289      3.840      0.000      0.057
    FRLUNCHJ        -2.164      1.635     -1.323      0.186      0.117
    CONDITION        2.385      0.652      3.656      0.000      0.074

 Intercepts
    PSOLVEPST       52.532      0.521    100.790      0.000      0.049

 Residual Variances
    PSOLVEPST        1.819      0.740      2.459      0.014      0.078
```

Mplus separates the level-1 and level-2 effects on the output (labeled `Within Level` and `Between Level`, respectively). The primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.39 points higher than control schools, on average, controlling for student- and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero ($z = 3.66$, $p < .001$). Note that these estimates are virtually identical to those from MCMC estimation in the next section.

16.8 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex16.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex16.spwb

```
CD '/users/username/desktop'.
DATA LIST free file =  'imps.dat'
  /imputation_ school student condition teachexp eslpct hispanic male
   frlunch lowach stanmath efficacypre efficacypst psolvepre psolvepst.
EXE.

AGGREGATE
  /outfile = * mode = addvariables overwrite = yes
  /break=imputation_ school
  /psolvepre_meanj = mean(psolvepre)
  /hispanic_meanj = mean(hispanic)
  /frlunch_meanj = mean(frlunch).
EXE.

AGGREGATE
  /outfile = * mode = addvariables overwrite = yes
  /psolvepre_mean = mean(psolvepre)
  /hispanic_mean = mean(hispanic)
  /frlunch_mean = mean(frlunch).
EXE.

COMPUTE psolvepre_cwc = psolvepre - psolvepre_meanj.
COMPUTE hispanic_cwc = hispanic - hispanic_meanj.
COMPUTE frlunch_cwc = frlunch - frlunch_meanj.
COMPUTE psolvepre_meanj = psolvepre_meanj - psolvepre_mean.
COMPUTE hispanic_meanj = hispanic_meanj - hispanic_mean.
COMPUTE frlunch_meanj = frlunch_meanj - frlunch_mean.
EXE.
```

The first line uses the CD command to change the working directory to the desktop. The username portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 7, the AGGREGATE command adds the grand means to the data, whereas the AGGREGATE command on line 15 adds the grand means into the data. Beginning on line 22, level-1 predictors are centered at their group means, and level-2 predictors are centered at their grand means.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 29 sorts the data by the imputation index variable, and the SPLIT FILE command on line 30 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 31.

SPSS Script Ex16.spwb, continued

```
29  SORT CASES by imputation_.
30  SPLIT FILE layered by imputation_.
31  MIXED psolvepst with psolvepre_cwc hispanic_cwc frlunch_cwc
32     psolvepre_meanj hispanic_meanj frlunch_meanj condition
33    /print = solution testcov
34    /fixed = intercept psolvepre_cwc hispanic_cwc frlunch_cwc
35     psolvepre_meanj hispanic_meanj frlunch_meanj condition
36    /random = intercept | subject(school) covtype(id).
```

16.9 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The output also includes pooled estimates of level-2 variance and covariance parameters. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The results are interpreted in the same way as a complete-data multilevel analysis. The pooled regression coefficients are in the table labeled `Estimates of Fixed Effects`. The primary focus is the γ_{04} coefficient, which indicates that intervention schools scored 2.39 points higher than control schools, on average, controlling for student-and school-level covariates. The corresponding test statistic indicates that the group mean difference is statistically different from zero ($t = 3.13$, $p < .001$). The pooled random intercept and within-cluster residual variances are in the table labeled `Estimates of Covariance Parameters`.

Estimates of Fixed Effects[a]

imputation_	Parameter	Estimate	Std. Error	df	t	Sig.	95% Confidence Interval Lower Bound	95% Confidence Interval Upper Bound	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	Intercept	52.580	.496	26.076	106.019	<.001	51.561	53.600			
	psolvepre_cwc	.467	.031	951.185	14.858	<.001	.405	.529			
	hispanic_cwc	1.338	.364	951.185	3.671	<.001	.623	2.053			
	frlunch_cwc	-.646	.396	951.185	-1.629	.104	-1.423	.132			
	psolvepre_meanj	.564	.211	22.764	2.676	.014	.128	1.000			
	hispanic_meanj	5.139	1.321	26.071	3.890	<.001	2.423	7.854			
	frlunch_meanj	-2.427	2.279	22.312	-1.065	.298	-7.149	2.296			
	condition	2.753	.674	24.411	4.084	<.001	1.363	4.142			
2.00	Intercept	52.722	.554	25.785	95.179	<.001	51.583	53.861			
	psolvepre_cwc	.439	.031	951.063	13.950	<.001	.377	.500			
	hispanic_cwc	.730	.367	951.063	1.986	.047	.009	1.450			
	frlunch_cwc	-.841	.403	951.063	-2.087	.037	-1.632	-.050			
	psolvepre_meanj	.645	.230	23.111	2.802	.010	.169	1.121			
	hispanic_meanj	5.079	1.452	25.190	3.497	.002	2.089	8.069			
	frlunch_meanj	-2.811	2.699	22.380	-1.041	.309	-8.404	2.782			
	condition	2.537	.754	24.428	3.367	.003	.984	4.091			
• • •											
20.00	Intercept	52.668	.508	25.853	103.726	<.001	51.624	53.712			
	psolvepre_cwc	.471	.031	951.009	15.263	<.001	.411	.532			
	hispanic_cwc	.729	.360	951.009	2.025	.043	.023	1.436			
	frlunch_cwc	-.627	.390	951.009	-1.609	.108	-1.391	.138			
	psolvepre_meanj	.675	.215	22.828	3.138	.005	.230	1.119			
	hispanic_meanj	5.230	1.309	25.503	3.997	<.001	2.537	7.922			
	frlunch_meanj	-1.522	2.408	22.697	-.632	.534	-6.507	3.464			
	condition	2.522	.694	24.260	3.635	.001	1.091	3.952			
Pooled	Intercept	52.713	.520		101.440	<.001	51.694	53.732	.055	.058	.997
	psolvepre_cwc	.457	.033		13.657	<.001	.392	.523	.126	.142	.994
	hispanic_cwc	.884	.432		2.044	.042	.032	1.736	.288	.394	.986
	frlunch_cwc	-.763	.448		-1.702	.090	-1.644	.118	.221	.278	.989
	psolvepre_meanj	.648	.215		3.017	.003	.227	1.069	.041	.042	.998
	hispanic_meanj	4.946	1.369		3.613	<.001	2.262	7.629	.051	.053	.997
	frlunch_meanj	-2.184	2.482		-.880	.379	-7.050	2.682	.051	.053	.997
	condition	2.385	.710		3.362	<.001	.994	3.777	.064	.068	.997

a. Dependent Variable: psolvepst.

Estimates of Covariance Parameters[a]

imputation_	Parameter		Estimate	Std. Error	Wald Z	Sig.	95% Confidence Interval Lower Bound	95% Confidence Interval Upper Bound	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	Residual		20.502	.940	21.808	<.001	18.740	22.430			
	Intercept [subject = school]	Variance	2.276	.839	2.711	.007	1.104	4.689			
2.00	Residual		20.552	.942	21.807	<.001	18.786	22.485			
	Intercept [subject = school]	Variance	2.885	1.016	2.841	.005	1.447	5.752			
• • •											
20.00	Residual		19.811	.909	21.806	<.001	18.108	21.674			
	Intercept [subject = school]	Variance	2.427	.879	2.761	.006	1.193	4.936			
Pooled	Residual		20.388	1.059		<.001	18.306	22.470	.225	.283	.989
	Intercept [subject = school]	Variance	2.313	.883		.009	.581	4.045	.085	.092	.996

a. Dependent Variable: psolvepst.

17

MCMC: Random Slope Model With an Interaction

This example illustrates a two-level multiple regression with random intercepts. The analysis uses the `problemsolving3level.dat` data set taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 928 student-level records include pretest and posttest math problem-solving and self-efficacy scores, standardized math scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
STUDENT	Student (level-2) identifier	0	Integer index
PROBSOLVE	Math problem-solving	11.4	Numeric (37 to 68)
MONTH	Time scores (initial = 0)	0	Numeric (0 to 6)
HISPANIC	Ethnicity/race	9.0	0 = Non-Hispanic, 1 = Hispanic
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Lunch assistance
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental

17.1 Analysis Model

The analysis is a linear growth model that features a repeatedly-measured problem-solving test regressed on time scores (months since the start of the school year, a level-1 predictor), experimental condition (level-2), the cross-level interaction of the two variables, and two grand mean centered student-level dummy codes (the Hispanic and lunch assistance indicators). To

convey each variable's level, the *i* and *j* subscripts denote repeated measurements and students, respectively. The combined regression model below.

$$\begin{aligned} PROBSOLVE_{ij} &= (\gamma_{00} + u_{0j}) + (\gamma_{10} + u_{1j})(MONTH_{ij}) \\ &+ \gamma_{11}(HISPANIC_j)(MONTH_{ij}) + \gamma_{12}(CONDITION_j)(MONTH_{ij}) \\ &+ \gamma_{01}(HISPANIC_j) + \gamma_{02}(FRLUNCH_{ij}^{cgm}) + \gamma_{03}(CONDITION_j) + \varepsilon_{ij} \end{aligned} \tag{31}$$

All coefficients with a leading zero subscript are determinants of baseline performance, and all coefficients with one as a leading subscript define the monthly change rates. In particular, γ_{11} is the degree to which ethnicity moderates the change rates, and γ_{12} captures the moderating effect of the intervention. The *cgm* superscript indicates centering at the grand mean. Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Blimp uses a factored regression specification that assigns separate distributions to the predictors and outcome. By default, Blimp invokes a multivariate normal distribution for numeric predictors and the latent response scores for discrete predictors.

17.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex17.1.inp`. The first six lines can be viewed as a set of commands that specify information about the data and variables. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex17.1.imp

```
DATA: problemsolving3level.dat;
VARIABLES: school student wave condition teachexp eslpct ethnic
  male frlunch lowach stanmath month month7 probsolve efficacy;
CLUSTERID: student;
ORDINAL: hispanic frlunch condition;
MISSING: 999;
FIXED: month condition;
CENTER: grandmean = frlunch;
MODEL:
month hispanic frlunch condition month*condition month*hispanic | month;
SIMPLE:
month | condition;
month | hispanic;
SEED: 90291;
```

```
15  BURN: 1000;
16  ITERATIONS: 20000;
```

The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The CLUSTERID command on line 4 lists the student-level identifier variable that indicates the clustering of the repeated measurements within students. Including the CLUSTERID command automatically introduces random intercepts. The ORDINAL command on line 5 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The MISSING command on line 6 defines a global missing value code as 999.

The FIXED, CENTER, MODEL, and SIMPLE blocks can be viewed as a set. The FIXED command identifies complete predictors that do not require a distribution or regression model. The CENTER command deviates a covariate at its iteratively-estimated grand mean. Beginning on line 9, the MODEL command lists the regression model, with outcome variable to the left of the tilde and predictors to the right. The product term is specified by joining the interacting variables with an asterisk (i.e., MONTH*CONDITION), and listing MONTH to the right of the vertical pipe specifies this variable as a random slope predictor. Starting on line 11, the SIMPLE command requests two sets of conditional effects (i.e., simple slopes) that give the effect of MONTH at each level of CONDITION and HISPANIC. By default, Blimp computes the simple slope at each level of a binary moderator listed on the ORDINAL line. Blimp automatically configures the explanatory variable models under the assumption that the numeric variables and latent response scores (discrete predictors) are normally distributed. Custom significance tests can be specified using the WALDTEST command, as shown in previous examples. Finally, lines 14 through 16 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex17.R is shown below.

rblimp Script Ex17.R

```
1  library(rblimp)
2  load('problemsolving3level.rda')
3
```

```
mymodel <- rblimp(
  data = problemsolving3level,
  clusterid = 'student',
  ordinal = 'hispanic frlunch condition',
  fixed = 'month condition',
  center = 'grandmean = frlunch',
  model = 'probsolve ~ month hispanic frlunch condition
    month*condition month*hispanic | month',
  simple = 'month | condition; month | hispanic',
  seed = 90291,
  burn = 10000,
  iter = 20000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp` function. The two exceptions are the `VARIABLES` and `MISSING` commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the `MODEL` command) are separated by semicolons, as they are in the Blimp script. Finally, the `output(mymodel)` function prints the Blimp output.

17.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          251 to 500             1.124            29
                          501 to 1000            1.182             3
                          751 to 1500            1.162             3
                                  ...              ...            ..
                         4501 to 9000            1.009             2
                         4751 to 9500            1.008            13
                         5001 to 10000           1.009            18
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3, but it additionally reports the number of observations at each level.

```
DATA INFORMATION:

  Level-2 identifier:       student
  Sample Size:              6874
  Level-2 Clusters:         982
  Missing Data Rates:

               probsolve = 11.45
                hispanic = 08.96
                 frlunch = 04.68
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals

describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 20000 iterations using 2 chains.


Outcome Variable:  probsolve

Grand Mean Centered: frlunch
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
L2 : Var(Intercept)	11.289	0.810	9.782	12.950	---	---	1382.218
L2 : Cov(month,Intercept)	0.039	0.118	-0.213	0.247	---	---	338.722
L2 : Var(month)	0.108	0.030	0.056	0.175	---	---	183.745
Residual Var.	12.567	0.274	12.043	13.119	---	---	1549.938
Coefficients:							
Intercept	49.364	0.306	48.770	49.974	26003.056	0.000	1802.836
month	0.274	0.060	0.159	0.392	21.261	0.000	6927.856
hispanic	1.359	0.300	0.766	1.945	20.531	0.000	1575.087
frlunch	-0.954	0.314	-1.566	-0.344	9.195	0.002	1278.934
condition	-0.420	0.280	-0.966	0.135	2.251	0.134	1712.019
month*condition	0.367	0.053	0.262	0.470	47.320	0.000	8135.002
month*hispanic	0.222	0.058	0.108	0.335	14.861	0.000	6372.419
Standardized Coefficients:							
month	0.103	0.022	0.060	0.147	21.165	0.000	6594.495
hispanic	0.120	0.026	0.068	0.171	21.025	0.000	1596.832
frlunch	-0.071	0.023	-0.116	-0.026	9.279	0.002	1281.246
condition	-0.038	0.026	-0.088	0.012	2.249	0.134	1712.357
month*condition	0.147	0.021	0.105	0.189	47.391	0.000	8432.508

```
  month*hispanic                    0.090     0.023     0.044     0.136    14.927     0.000  6528.861

Proportion Variance Explained
  by Coefficients                   0.106     0.010     0.088     0.127       ---       ---  1591.477
  by Level-2 Random Intercepts      0.437     0.015     0.409     0.467       ---       ---  5856.196
  by Level-2 Random Slopes          0.015     0.004     0.008     0.024       ---       ---   183.785
  by Level-1 Residual Variation     0.440     0.013     0.415     0.466       ---       ---  2544.204

                                  ------------------------------------------------------------------------
```

To begin, the `N_Eff` values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the `N_Eff` values are insufficient, increasing the value on the `ITERATIONS` command will remedy the issue. Unlike previous examples, this analysis specified 20,000 iterations because the effective sample size for the random slope variance was less than 100 when using 10,000 iterations.

The results are interpreted in the same way as a complete-data multilevel analysis. The first section of the output table displays the variance estimates. The random intercept and slope variances are denoted `L2:Var(Intercept)` and `L2:Var(month)`, respectively, and their covariance is labeled `L2 : Cov(month,Intercept)`. The within-cluster residual variance is denoted `Residual Var.` Turning to the coefficients section, lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of `MONTH` (γ_{10} = 0.27) is the monthly change rate for non-Hispanic students (`HISPANIC` = 0) in the comparison condition (`CONDITION` = 0). The intervention mean difference (γ_{03} = −0.42) similarly reflects the mean difference when `MONTH` = 0 (at the first assessment). One cross-level interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient (γ_{12} = 0.37) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The other interaction captures the growth rate difference for Hispanic students. The positive coefficient (γ_{11} = 0.22) indicates that the growth rate for the Hispanic students is greater (more positive) than that of non-Hispanics. The 95% credible interval limits suggest that both interaction effects are statistically different from zero (p < .05) because the null value is outside the interval. The frequentist test statistics and p-values give the same conclusion. Finally, the bottom section of the table displays Rights and Sterba (2019) R-squared effect size values. The fixed effects explain 10.6% of the total variation, the random intercepts account for 43.7% of the variability, and the random slopes account for 1.5% of the variation.

The SIMPLE command prints a table of conditional effects (simple slopes) of MONTH within each intervention condition and ethnicity group. Consistent with the positive interaction coefficients, the simple slopes for the experimental schools and Hispanics are higher (more positive) than the growth rates for controls and non-Hispanics. All four conditional effects are statistically significant at $p < .05$ because the null value does not fall within the 95% credible intervals. The output table is shown below.

```
Conditional Effects              Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
  month | condition @ 0
    Intercept                    49.364      0.306     48.770     49.974  26003.056      0.000   1802.836
    Slope                         0.274      0.060      0.159      0.392     21.261      0.000   6927.856

  month | condition @ 1
    Intercept                    48.945      0.263     48.434     49.463  34749.384      0.000   1572.191
    Slope                         0.641      0.048      0.547      0.737    175.765      0.000   7697.318

  month | hispanic @ 0
    Intercept                    49.364      0.306     48.770     49.974  26003.056      0.000   1802.836
    Slope                         0.274      0.060      0.159      0.392     21.261      0.000   6927.856

  month | hispanic @ 1
    Intercept                    50.724      0.235     50.269     51.191  46749.578      0.000   1747.997
    Slope                         0.497      0.045      0.409      0.586    121.749      0.000   8389.335

-------------------------------------------------------------------------------------------------------

NOTE: Intercepts are computed by setting all predictors
      not involved in the conditional effect to zero.
```

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in OUTCOME MODEL ESTIMATES section with the focal results, and the auto-generated predictor models are displayed under the heading PREDICTOR MODEL ESTIMATES. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

17.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (10,000 sets in this example). A subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The

Blimp input file `Ex17.2.imp` is identical `Ex17.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     school student wave condition teachexp eslpct ethnic male
     frlunch lowach stanmath month0 month7 probsolve efficacy
```

```
stacked = './imps/imps.dat'

  imp# school student wave condition teachexp eslpct ethnic male
  frlunch lowach stanmath month0 month7 probsolve efficacy
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex17.R

```
library(rblimp)
load('problemsolving3level.rda')

mymodel <- rblimp(
  data = problemsolving3level,
  clusterid = 'student',
  ordinal = 'hispanic frlunch condition',
  fixed = 'month condition',
  center = 'grandmean = frlunch',
  model = 'probsolve ~ month hispanic frlunch condition
    month*condition month*hispanic | month',
  simple = 'month | condition; month | hispanic',
  seed = 90291,
  burn = 10000,
  iter = 20000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple

imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

17.5 Analyzing Multiple Imputations in R

Continuing with the previous `rblimp` script, the following excerpt from `Ex7.R` shows how to perform multiple imputation inference. The script requires the `mitml` package (Grund et al., 2023).

R Script Ex17.R

```
1   library(rblimp)
2   library(lme4)
3   library(mitml)
4   load('problemsolving3level.rda')
5
6   mymodel <- rblimp(...)
7
8   implist <- as.mitml(mymodel)
9
10  mean_frlunch <- mean(unlist(lapply(implist, function(df) mean(df$frlunch))))
11  for (i in 1:length(implist)) {
12    implist[[i]]$frlunch.cgm <- implist[[i]]$frlunch - mean_frlunch
13  }
14
15  fit <- with(implist,
16    lmer(probsolve ~ month + frlunch + hispanic + condition
17    + month:condition + month:hispanic + (1 + month | student), REML = T))
18
19  estimates <- testEstimates(fit, extra.pars = T)
20  estimates
21  confint(estimates)
```

To begin, `as.mitml` on Line 8 is an `rblimp` function that converts the imputation object into a list of data sets called `implist`, as required by the `mitml` package. Lines 10 through 13 center the covariate at its grand mean. Lines 15 through 17 fit the focal regression model using the `lmer` function, and line 19 uses the `testEstimates` function in `mitml` to implement Rubin's pooling

rules and save the results in an object called `estimates`. Lines 20 and 21 print the pooled estimates and confidence intervals.

Following a significant group-by-time interaction effect, researchers typically examine the slope of the focal predictor at different values of the moderator. The final code block below computes these conditional effects or simple slopes of the monthly change rate at each value of `CONDITION` and `HISPANIC`. The `constraints` parameter is a text string that defines the computation of the conditional growth rate in each subgroup.

R Script Ex17.R, continued

```
22  testConstraints(fit, constraints = 'month + month*condition*0')
23  testConstraints(fit, constraints = 'month + month*condition*1')
24  testConstraints(fit, constraints = 'month + month*hispanic*0')
25  testConstraints(fit, constraints = 'month + month*hispanic*1')
```

17.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

Considering the coefficients, lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of `MONTH` ($\hat{\gamma}_{10}$ = 0.27) is the monthly change rate for non-Hispanic students (`HISPANIC` = 0) in the comparison condition (`CONDITION` = 0). The intervention mean difference ($\hat{\gamma}_{03}$ = −0.42) similarly reflects the mean difference when `MONTH` = 0 (at the first assessment). One cross-level interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\gamma}_{12}$ = 0.37) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The other interaction captures the growth rate difference for Hispanic students. The positive coefficient ($\hat{\gamma}_{11}$ = 0.22) indicates that the growth rate for the Hispanic students is greater (more positive) than that of non-Hispanics. The corresponding test statistics indicate that both interaction effects are statistically different from

zero ($p < .001$). Note that these estimates are numerically equivalent to those from MCMC estimation.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)       49.367     0.309   159.532  1677.585     0.000     0.119     0.107
month              0.269     0.061     4.437   330.193     0.000     0.316     0.244
frlunch.cgm       -0.923     0.314    -2.939  2950.749     0.003     0.087     0.081
hispanic           1.365     0.294     4.644  2516.400     0.000     0.095     0.088
condition         -0.417     0.278    -1.499  9015.743     0.134     0.048     0.046
month:condition    0.374     0.053     7.042   867.074     0.000     0.174     0.150
month:hispanic     0.220     0.056     3.945   656.403     0.000     0.205     0.173

                            Estimate
Intercept~~Intercept|student   11.171
month~~month|student            0.107
Intercept~~month|student        0.053
Residual~~Residual             12.561
ICC|student                     0.471

                    2.5 %     97.5 %
(Intercept)     48.7600507 49.9739434
month            0.1495981  0.3878664
frlunch.cgm     -1.5389275 -0.3072803
hispanic         0.7886026  1.9411189
condition       -0.9627592  0.1282146
month:condition  0.2695923  0.4779414
month:hispanic   0.1103965  0.3291800
```

Finally, the printed output also includes the table of conditional effects or simple slopes. Consistent with the positive interaction coefficient, the monthly growth rate for the experimental schools is higher (more positive) that the growth rate for controls.

```
Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:
```

```
                               Estimate Std. Error
   month + month*condition*0:      0.269      0.061

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     19.690       1 224.632   0.000   0.316

Unadjusted hypothesis test as appropriate in larger samples.


Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                               Estimate Std. Error
   month + month*condition*1:      0.157      0.090

Combination method: D1

     F.value      df1      df2    P(>F)      RIV
       3.022        1 2026.971    0.082    0.084

Unadjusted hypothesis test as appropriate in larger samples.

Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                              Estimate Std. Error
   month + month*hispanic*0:      0.269      0.061

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     19.690       1 224.632   0.000   0.316

Unadjusted hypothesis test as appropriate in larger samples.


Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:
```

```
                              Estimate Std. Error
   month + month*hispanic*1:      0.636      0.181

Combination method: D1

    F.value     df1     df2   P(>F)     RIV
     12.375       1 397.963   0.000   0.217

Unadjusted hypothesis test as appropriate in larger samples.
```

17.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). The contents of this file were shown in Section 12.6. The Mplus input file for analyzing the imputations is Ex17.inp.

Mplus Script Ex17.inp

```
1   DATA:
2   file = ./imps/implist.dat;
3   type = imputation;
4   VARIABLE:
5   names = school student wave condition teachexp eslpct ethnic
6     male frlunch lowach stanmath month0 month7 probsolve efficacy;
7   usevariables = probsolve month hispanic frlunch condition;
8   cluster = student;
9   within = month male frlunch;
10  between = hispanic frlunch condition;
11  DEFINE:
12  center frlunch (grandmean);
13  ANALYSIS:
14  type = twolevel random;
15  MODEL:
16  %within%
```

```
17  ranslope | probsolve on month;
18  %between%
19  [ranslope] (beta1);
20  probsolve on hispanic frlunch condition;
21  ranslope on hispanic condition (beta5-beta6)
22  ranslope with probsolve;
23  MODEL CONSTRAINT:
24  new(slp_c0h0 slp_hisp1 slp_cond1);
25  slp_c0h0 = beta1;
26  slp_hisp1 = beta1 + beta5;
27  slp_cond1 = beta1 + beta6;
28  OUTPUT:
29  cinterval;
```

The DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand instructs Mplus that the input data is a list of file names. The VARIABLE command provides information about the data. Beginning on line 5, the names subcommand assigns names to the variables in the input data file, and the usevariables subcommand selects variables for the analysis. The cluster command on line 8 lists the school-level identifier variable that indicates the clustering of the data records in schools. The within and between subcommands on lines 9 and 10 identify level-1 and level-2 predictors, respectively.

On line 12, the center subcommand under the DEFINE command centers the three covariates at their grand means. The ANALYSIS command and type = twolevel random subcommand is required for estimating two-level models with random slopes. The MODEL section of the script consists of two sections: the %within% section specifies the regression of the outcome on level-1 predictors, and the %between% section specifies the regression of the random intercepts on the level-2 predictors. In the %within% section, listing ranslope (an arbitrary name) to the left of the vertical pipe creates a level-2 latent variable capturing individual growth rates. Regressing this latent variable on CONDITION in the %between% model gives the cross-level interaction. Beginning on line 23, the MODEL CONSTRAINT command is used to compute conditional effects or simple slopes. First, line 24 assigns names to three new parameters (the group-specific growth rates). Lines 25 through 27 use parameter labels from the MODEL section to compute the conditional effect of MONTH7 in each experimental group.

17.8 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing

Within Level

Residual Variances
    PROBSOLVE       12.561      0.353     35.534      0.000      0.091

Between Level

 RANSLOPE   ON
    HISPANIC         0.220      0.055      3.971      0.000      0.175
    CONDITION        0.374      0.052      7.166      0.000      0.155

 PROBSOLVE  ON
    HISPANIC         1.365      0.293      4.656      0.000      0.088
    FRLUNCH         -0.923      0.304     -3.033      0.002      0.086
    CONDITION       -0.417      0.276     -1.513      0.130      0.047

 RANSLOPE WITH
    PROBSOLVE        0.057      0.124      0.454      0.650      0.265

 Intercepts
    PROBSOLVE       49.367      0.304    162.404      0.000      0.112
    RANSLOPE         0.269      0.059      4.559      0.000      0.258

 Residual Variances
    PROBSOLVE       11.104      0.803     13.833      0.000      0.083
    RANSLOPE         0.105      0.033      3.193      0.001      0.294

New/Additional Parameters
```

```
SLP_C0H0          0.269      0.059       4.559      0.000      0.258
SLP_HISP          0.489      0.043      11.287      0.000      0.201
SLP_COND          0.643      0.047      13.575      0.000      0.131
```

Mplus separates the level-1 and level-2 effects on the output (labeled `Within Level` and `Between Level`, respectively). Considering the coefficients, lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of `MONTH` ($\hat{\gamma}_{10} = 0.27$) is the monthly change rate for non-Hispanic students (`HISPANIC` = 0) in the comparison condition (`CONDITION` = 0). The intervention mean difference ($\hat{\gamma}_{03} = -0.42$) similarly reflects the mean difference when `MONTH` = 0 (at the first assessment). One cross-level interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\gamma}_{12} = 0.37$) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The other interaction captures the growth rate difference for Hispanic students. The positive coefficient ($\hat{\gamma}_{11} = 0.22$) indicates that the growth rate for the Hispanic students is greater (more positive) than that of non-Hispanics. The corresponding test statistics indicate that both interaction effects are statistically different from zero ($p < .001$). Note that these estimates are numerically equivalent to those from MCMC estimation.

17.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex17.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex17.spwb

```
1  CD '/users/username/desktop'.
2  DATA LIST free file =  'imps.dat'
3    /imputation_ school student wave condition teachexp eslpct hispanic
4     male frlunch lowach stanmath month month7 probsolve efficacy.
5  EXE.
6
7  AGGREGATE
```

```
8   /outfile = * mode = addvariables overwrite = yes
9   /frlunch_mean = mean(frlunch).
10  EXE.
11
12  COMPUTE frlunch_cgm = frlunch - frlunch_mean.
13  EXE.
```

The first line uses the CD command to change the working directory to the desktop. The username portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 7, the AGGREGATE command adds the grand mean to the data. On line 12, a new variable is created that centers the FRLUNCH predictor at its pooled grand mean.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 14 sorts the data by the imputation index variable, and the SPLIT FILE command on line 15 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 16.

SPSS Script Ex17.spwb, continued

```
14  SORT CASES by imputation_.
15  SPLIT FILE layered by imputation_.
16  MIXED probsolve with month frlunch_cgm hispanic condition
17    /print = solution testcov
18    /fixed = month frlunch_cgm hispanic condition
19     hispanic*condition month*condition
20    /random = intercept month | subject(student) covtype(un).
```

17.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics.

The output also includes pooled estimates of level-2 variance and covariance parameters. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The pooled regression coefficients are in the table labeled `Estimates of Fixed Effects`. The results are interpreted in the same way as a complete-data multilevel analysis. Lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of `MONTH` ($\hat{\gamma}_{10} = 0.43$) is the monthly change rate for non-Hispanic students (`HISPANIC` = 0) in the comparison condition (`CONDITION` = 0). The intervention mean difference ($\hat{\gamma}_{03} = -0.54$) similarly reflects the mean difference when `MONTH` = 0 (at the first assessment). One cross-level interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\gamma}_{12} = 0.36$) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The other interaction captures the growth rate difference for Hispanic students. The positive coefficient ($\hat{\gamma}_{11} = 0.24$) indicates that the growth rate for the Hispanic students is greater (more positive) than that of non-Hispanics. The test statistic corresponding to the interaction between the `MONTH` and `CONDITION` indicates that this interaction is significant ($p < .001$), whereas the `HISPANIC` and `CONDITION` interaction was nonsignificant ($p = .68$). The pooled variance–covariance matrix of the random effects and the within-cluster variance are in the table labeled `Estimates of Covariance Parameters`.

Estimates of Fixed Effects[a]

imputation_	Parameter	Estimate	Std. Error	df	t	Sig.	95% Confidence Interval		Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
							Lower Bound	Upper Bound			
1.00	Intercept	49.102575	.372571	1067.852	131.794	.000	48.371521	49.833630			
	month	.434736	.038773	982.000	11.212	.000	.358647	.510824			
	frlunch_cgm	-.955896	.299979	982.000	-3.187	.001	-1.544570	-.367221			
	hispanic	1.602819	.425211	982.000	3.769	.000	.768392	2.437247			
	condition	-.605156	.454661	1071.982	-1.331	.183	-1.497282	.286970			
	hispanic * condition	.516434	.528792	982.000	.977	.329	-.521259	1.554127			
	month * condition	.338255	.049521	982.000	6.830	.000	.241075	.435434			
2.00	Intercept	49.155520	.374731	1060.911	131.175	.000	48.420222	49.890819			
	month	.395372	.039254	982.000	10.072	.000	.318340	.472404			
	frlunch_cgm	-.847165	.297662	982.000	-2.846	.005	-1.431292	-.263037			
	hispanic	1.700290	.427963	982.000	3.973	.000	.860464	2.540117			
	condition	-.676620	.457116	1064.832	-1.480	.139	-1.573571	.220331			
	hispanic * condition	.386599	.531993	982.000	.727	.468	-.657374	1.430573			
	month * condition	.396776	.050135	982.000	7.914	.000	.298391	.495160			
...											
20.00	Intercept	48.992897	.375527	1061.391	130.464	.000	48.256037	49.729757			
	month	.418570	.038578	982.000	10.850	.000	.342866	.494274			
	frlunch_cgm	-.820867	.303299	982.000	-2.706	.007	-1.416057	-.225678			
	hispanic	1.879289	.429429	982.000	4.376	.000	1.036584	2.721993			
	condition	-.304944	.458672	1065.249	-.665	.506	-1.204947	.595060			
	hispanic * condition	-.086646	.534367	982.000	-.162	.871	-1.135279	.961986			
	month * condition	.357953	.049271	982.000	7.265	.000	.261264	.454642			
Pooled	Intercept	49.108628	.399230		123.008	.000	48.325256	49.892001	.136	.155	.993
	month	.426440	.043437		9.817	.000	.341058	.511821	.216	.270	.989
	frlunch_cgm	-.922830	.313595		-2.943	.003	-1.537725	-.307935	.082	.089	.996
	hispanic	1.725232	.454848		3.793	.000	.832846	2.617619	.127	.144	.994
	condition	-.540689	.484923		-1.115	.265	-1.491955	.410577	.119	.133	.994
	hispanic * condition	.236195	.577000		.409	.682	-.896440	1.368831	.157	.184	.992
	month * condition	.356721	.052855		6.749	.000	.253010	.460431	.134	.153	.993

a. Dependent Variable: probsolve.

Estimates of Covariance Parameters[a]

imputation_	Parameter		Estimate	Std. Error	Wald Z	Sig.	95% Confidence Interval Lower Bound	95% Confidence Interval Upper Bound	Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
1.00	Residual		12.492379	.252127	49.548	.000	12.007865	12.996443			
	Intercept + month [subject = student]	UN (1,1)	11.180628	.778452	14.363	.000	9.754415	12.815371			
		UN (2,1)	-.005460	.112582	-.048	.961	-.226117	.215197			
		UN (2,2)	.125128	.027309	4.582	.000	.081579	.191923			
• • •											
20.00	Residual		12.521608	.252717	49.548	.000	12.035961	13.026852			
	Intercept + month [subject = student]	UN (1,1)	11.237643	.780194	14.404	.000	9.807967	12.875718			
		UN (2,1)	.048177	.111309	.433	.665	-.169985	.266338			
		UN (2,2)	.118328	.027071	4.371	.000	.075570	.185278			
Pooled	Residual		12.560644	.274746		.000	12.021394	13.099894	.151	.175	.993
	Intercept + month [subject = student]	UN (1,1)	11.164051	.812605		.000	9.570699	12.757403	.082	.089	.996
		UN (2,1)	.030114	.129172		.816	-.224092	.284320	.258	.338	.987
		UN (2,2)	.115894	.032482		.000	.051847	.179942	.313	.442	.985

a. Dependent Variable: probsolve.

18

MCMC: Three-Level Growth Model

This example illustrates a two-level multiple regression with random intercepts. The analysis uses the `problemsolving3level.dat` data set taken from a cluster-randomized educational intervention where 29 schools were assigned to an intervention and comparison condition. In addition to the intervention assignment indicator, school-level variables include the average years of teacher experience and the percentage of learners for whom English is a second language. The 928 student-level records include pretest and posttest math problem-solving and self-efficacy scores, standardized math scores taken from a statewide assessment, and several sociodemographic variables. The analysis variables are as follows.

Name	Definition	Missing %	Scale
	Identifier Variables		
SCHOOL	School identifier	0	Integer index
STUDENT	Student identifier	0	Integer index
	Focal Variables		
PROBSOLVE	Math problem-solving posttest	11.5	Numeric
$MONTH_7$	Time scores (end of year = 0)	0	Numeric (–6 to 0)
MALE	Gender dummy code	0	0 = Female, 1 = Male
FRLUNCH	Lunch assistance code	4.7	0 = None, 1 = Free/reduced lunch
TEACHEXP	Teacher years of experience	10.8	Numeric
CONDITION	Experimental condition	0	0 = Control, 1 = Experimental

18.1 Analysis Model

The analysis is a linear growth model that features a repeatedly-measured problem-solving test regressed on time scores (months until the end of the school year, a level-1 predictor), experimental condition (level-2), the cross-level interaction of the two variables, and three grand mean centered covariates: gender and lunch assistance dummy codes (level-1), and years of teacher experience (level-2). To convey each variable's level, the *i* and *j* subscripts denote repeated measurements and students, respectively, and *k* is the school-level identifier.

$$\begin{aligned} PROBSOLVE_{ijk} = (\gamma_{00} + u_{0jk} + u_{0k}) + (\gamma_{10} + u_{1jk} + u_{1k})(MONTH_{7ij}) + \gamma_{01}(MALE_j^{cgm}) \\ + \gamma_{02}(FRLUNCH_j^{cgm}) + \gamma_{03}(TEACHEXP_j^{cgm}) + \gamma_{04}(CONDITION_j) \\ + \gamma_{11}(MONTH_{7ij})(CONDITION_j) + \varepsilon \end{aligned} \quad (32)$$

All coefficients with a leading zero subscript are determinants of end-of-year performance (the intercept, `MONTH7 = 0`), and all coefficients with one as a leading subscript define the monthly change rates. In particular, γ_{11} is the degree to which the intervention moderates the change rates. The *cgm* superscript indicates centering at the grand mean. Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors. Blimp uses a factored regression specification that assigns separate distributions to the predictors and outcome. By default, Blimp invokes a multivariate normal distribution for numeric predictors and the latent response scores for discrete predictors.

18.2 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex18.1.inp`. The first six lines can be viewed as a set of commands that specify information about the data and variables. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex18.1.imp

```
1  DATA: problemsolving3level.dat;
2  VARIABLES: school student wave condition teachexp eslpct ethnic
3    male frlunch lowach stanmath month0 month7 probsolve efficacy;
4  CLUSTERID: student school;
5  ORDINAL: male frlunch condition;
6  MISSING: 999;
```

```
7   FIXED: month7 male condition;
8   CENTER: grandmean = male frlunch teachexp;
9   MODEL:
10  probsolve ~ month7 male frlunch teachexp condition
11    month7*condition  | month7;
12  SIMPLE: month7 | condition;
13  SEED: 90291;
14  BURN: 20000;
15  ITERATIONS: 50000;
```

The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns. The CLUSTERID command on line 4 lists the student- and school-level identifier variables that indicates the clustering of the data records. The order of the identifier variables does not matter. Including the CLUSTERID command automatically introduces random intercepts at level-2 and level-3. The ORDINAL command on line 5 identifies binary and ordinal variables. Binary variables can be defined as ordinal or nominal, as the statistical models are identical. The MISSING command on line 6 defines a global missing value code as 999.

The FIXED, CENTER, MODEL, and SIMPLE blocks can be viewed as a set. The FIXED command identifies a complete predictor, which does not require a distribution or regression model. The CENTER command deviates the three covariates at their iteratively-estimated grand means. Beginning on line 9, the MODEL command lists the regression model, with outcome variable to the left of the tilde and predictors to the right. The product term is specified by joining the interacting variables with an asterisk (i.e., MONTH7*CONDITION), and listing MONTH7 to the right of the vertical pipe specifies this variable as a random slope predictor. The SIMPLE command requests the conditional effects (i.e., simple slopes) of MONTH7 at each level of CONDITION. By default, Blimp computes the simple slope at each level of a binary moderator listed on the ORDINAL line. Blimp automatically configures the explanatory variable models under the assumption that the numeric variables and latent response scores (discrete predictors) are normally distributed. Custom significance tests can be specified using the WALDTEST command, as shown in previous examples.

Finally, lines 13 through 15 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the

analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex18.R is shown below.

rblimp Script Ex18.R

```
library(rblimp)
load('problemsolving3level.rda')

mymodel <- rblimp(
  data = problemsolving3level,
  clusterid = 'school student',
  ordinal = 'male frlunch condition',
  fixed = 'month7 male condition',
  center = 'grandmean = male frlunch teachexp',
  model = 'probsolve ~ month7 male frlunch teachexp
    condition month7*condition | month7',
  simple = 'month7 | condition',
  seed = 90291,
  burn = 20000,
  iter = 50000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script. Finally, the output(mymodel) function prints the Blimp output.

18.3 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR

diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period. This analysis required a much longer burn-in period that previous examples.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                          501 to 1000            1.938            18
                         1001 to 2000            1.460             3
                         1501 to 3000            1.159             8
                                  ...              ...            ..
                         9001 to 18000           1.019            23
                         9501 to 19000           1.020            23
                        10001 to 20000           1.010            27
```

The next section of the output displays information about the variables in the analysis and the models used for estimation. This output table mimics the one from Section 6.3, but it additionally reports the number of observations at each level.

```
DATA INFORMATION:

  Level-2 identifier:       student
  Level-3 identifier:       school
  Sample Size:              6874
  Level-2 Clusters:         982
  Level-3 Clusters:         29
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of

the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 50000 iterations using 2 chains.

Outcome Variable:  probsolve

Grand Mean Centered: frlunch male teachexp
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
L2 : Var(Intercept)	11.082	0.848	9.535	12.841	---	---	1038.348
L2 : Cov(month7,Intercept)	0.322	0.122	0.098	0.577	---	---	285.124
L2 : Var(month7)	0.048	0.024	0.010	0.104	---	---	140.169
L3 : Var(Intercept)	7.701	2.876	4.243	15.322	---	---	6049.874
L3 : Cov(month7,Intercept)	0.643	0.294	0.282	1.412	---	---	9861.675
L3 : Var(month7)	0.094	0.039	0.047	0.198	---	---	12433.005
Residual Var.	12.569	0.268	12.049	13.102	---	---	1308.691
Coefficients:							
Intercept	52.887	0.822	51.230	54.495	4139.747	0.000	324.105
month7	0.455	0.098	0.263	0.649	21.531	0.000	1033.466
male	0.339	0.227	-0.102	0.787	2.217	0.137	3898.624
frlunch	-0.283	0.308	-0.881	0.325	0.845	0.358	3440.410
teachexp	0.000	0.001	-0.002	0.003	0.032	0.858	321.552
condition	1.551	1.100	-0.595	3.724	1.981	0.159	315.515

```
  month7*condition                0.297      0.131      0.031      0.549      5.100      0.024    939.274

Standardized Coefficients:
  month7                          0.167      0.036      0.095      0.237     21.338      0.000   1026.031
  male                            0.031      0.021     -0.009      0.071      2.217      0.136   3904.818
  frlunch                        -0.021      0.022     -0.064      0.024      0.846      0.358   3436.749
  teachexp                        0.009      0.067     -0.117      0.147      0.032      0.858    322.860
  condition                       0.139      0.096     -0.053      0.322      2.046      0.153    316.432
  month7*condition                0.116      0.051      0.012      0.212      5.199      0.023    978.851

Proportion Variance Explained
  by Coefficients                 0.072      0.018      0.045      0.116        ---        ---    450.110
  by Level-2 Random Intercepts    0.323      0.024      0.271      0.364        ---        ---   2792.166
  by Level-2 Random Slopes        0.006      0.003      0.001      0.014        ---        ---    140.792
  by Level-3 Random Intercepts    0.157      0.045      0.094      0.269        ---        ---   5603.249
  by Level-3 Random Slopes        0.013      0.005      0.006      0.025        ---        ---  12072.560
  by Level-1 Residual Variation   0.423      0.028      0.359      0.469        ---        ---   1665.135

------------------------------------------------------------------------------
```

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue. Unlike previous examples, this analysis specified 20,000 iterations because the effective sample size for the random slope variance was less than 100 when using 10,000 iterations. Unlike previous examples, this analysis specified 50,000 iterations to achieve acceptable values.

The results are interpreted in the same way as a complete-data multilevel analysis. The first section of the output table displays the variance estimates. The level-2 random intercept and slope variances are denoted L2:Var(Intercept) and L2:Var(month7), respectively, and their covariance is labeled L2 : Cov(month7,Intercept). Similarly, the level-3 random intercept and slope variances are denoted L3:Var(Intercept) and L3:Var(month7), respectively, and their covariance is labeled L3 : Cov(month7,Intercept). The within-cluster residual variance is denoted Residual Var. Turning to the coefficients section, lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of MONTH7 (γ_{10} = 0.46) is the monthly change rate for students in the comparison condition (CONDITION = 0), and the intervention slope (γ_{04} = 1.55) similarly reflects the mean difference when MONTH7 = 0 (at the final assessment). The interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient (γ_{11} = 0.30) indicates

that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. Finally, the bottom section of the table displays Rights and Sterba (2019) R-squared effect size values. The fixed effects explain 7.2% of the total variation, the random intercepts at level-2 and level-3 account for 32.3% and 15.7% of the variability, respectively, and the level-2 and level-3 random slopes account for 0.6% and 1.3% of the variation.

The `SIMPLE` command prints a table of conditional effects (simple slopes) of `MONTH7` within each intervention condition. Consistent with the positive interaction coefficient, the simple slope for the experimental schools is higher (more positive) that the growth rate for controls. Both conditional effects are statistically significant at $p < .05$ because the null value does not fall within the 95% credible intervals. The output table is shown below.

```
Conditional Effects                Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------

  month7 | condition @ 0
    Intercept                      52.887      0.822     51.230     54.495   4139.747      0.000    324.105
    Slope                           0.455      0.098      0.263      0.649     21.531      0.000   1033.466

  month7 | condition @ 1
    Intercept                      54.466      0.729     52.973     55.886   5576.054      0.000    258.252
    Slope                           0.749      0.086      0.579      0.915     76.330      0.000    781.490

-------------------------------------------------------------------------------------------------------

NOTE: Intercepts are computed by setting all predictors
      not involved in the conditional effect to zero.
```

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

18.4 Saving Model-Based Multiple Imputations

MCMC estimation imputes missing values at every iteration, such that the resulting Bayesian estimates average over thousands of plausible replacement scores (50,000 sets in this example). A

subset of the imputations can be saved for reanalysis in the frequentist framework, if desired. The Blimp input file `Ex18.2.imp` is identical `Ex18.1.imp`, but it adds the following lines at the bottom of the script.

```
NIMPS: 20;
CHAINS: 20;
SAVE:
stacked = ./imps/imps.dat;
separate = ./imps/imp*.dat;
```

The `NIMPS`, `CHAINS` and `SAVE` commands can be viewed as a set. Setting `NIMPS` equal to `CHAINS` saves a single filled-in data set from the final iteration of a unique `MCMC` process, thus avoiding autocorrelation among the imputations. The `SAVE` command provides a name for the imputed data sets. The script illustrates how to save data sets in two common formats. The stacked keyword creates a stacked file where all imputations are in a single file, and the separate keyword saves each imputed data set to a separate file with the asterisk replaced by a numeric index. To keep things organized, the `./imps` part of the file path points to a subfolder named `imps` located within the same folder as the script and data. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called `implist.dat` located in the `imps` folder).

When saving imputations, the bottom of the Blimp output file displays a table listing the order of the variables in the output data sets. All variables are saved regardless of whether they appeared in the fitted models. When saving data to a stacked file (e.g., for analysis in R or other packages), the first variable in the file is an integer index that identifies which data set each row belongs to (e.g., an integer variable that ranges from 1 to 20 in this example).

```
VARIABLE ORDER IN IMPUTED DATA:

  separate = './imps/imp*.dat'

     school student wave condition teachexp eslpct ethnic male
     frlunch lowach stanmath month0 month7 probsolve efficacy

   stacked = './imps/imps.dat'
```

```
imp# school student wave condition teachexp eslpct ethnic male
frlunch lowach stanmath month0 month7 probsolve efficacy
```

The imputed data sets are subsequently analyzed in another software package, and estimates and standard errors are combined using Rubin's rules (Little & Rubin, 2020). The analysis phase does not utilize the auxiliary variables, as their information is embedded in the imputations. Scripts for analyzing the imputed data sets are found in the next subsections.

In `rblimp`, the `NIMPS` and `CHAINS` commands are added as input parameters to the function as follows.

rblimp Script Ex18.R

```
library(rblimp)
load('problemsolving3level.rda')

mymodel <- rblimp(
  data = problemsolving3level,
  clusterid = 'school student',
  ordinal = 'male frlunch condition',
  fixed = 'month7 male condition',
  center = 'grandmean = male frlunch teachexp',
  model = 'probsolve ~ month7 male frlunch teachexp
    condition month7*condition | month7',
  simple = 'month7 | condition',
  seed = 90291,
  burn = 20000,
  iter = 50000,
  nimps = 20,
  chains = 20)
output(mymodel)
```

The `SAVE` command is no longer necessary because imputations are automatically stored in a `rblimp` object called `mymodel@imputations`. The next sections show how to analyze the multiple imputations. The multiple imputation point estimates, standard errors, and test statistics will be numerically equivalent to those produced by MCMC.

18.5 Analyzing Multiple Imputations in R

Continuing with the previous rblimp script, the following excerpt from Ex7.R shows how to perform multiple imputation inference. The script requires the mitml package (Grund et al., 2023).

R Script Ex18.R

```
1   library(rblimp)
2   library(lme4)
3   library(mitml)
4   load('problemsolving3level.rda')
5
6   mymodel <- rblimp(...)
7
8   implist <- as.mitml(mymodel)
9
10  mean_male <- mean(unlist(lapply(implist, function(df) mean(df$male))))
11  mean_frlunch <- mean(unlist(lapply(implist, function(df) mean(df$frlunch))))
12  mean_teachexp <- mean(unlist(lapply(implist, function(df) mean(df$teachexp))))
13  for (i in 1:length(implist)) {
14    implist[[i]]$male.cgm <- implist[[i]]$male - mean_male
15    implist[[i]]$frlunch.cgm <- implist[[i]]$frlunch - mean_frlunch
16    implist[[i]]$teachexp.cgm <- implist[[i]]$teachexp - mean_teachexp
17  }
18
19  fit <- with(implist,
20    lmer(probsolve ~ month7  + male.cgm + frlunch.cgm + teachexp.cgm
21    + condition + month7:condition + (1 + month7 | school/student), REML = T))
22
23  estimates <- testEstimates(fit, extra.pars = T)
24  estimates
25  confint(estimates)
```

To begin, as.mitml on Line 8 is an rblimp function that converts the imputation object into a list of data sets called implist, as required by the mitml package. Lines 10 through 17 center the covariates at at their pooled grand means. Lines 19 through 21 fit the focal regression model using the lmer function, and line 23 uses the testEstimates function in mitml to implement

Rubin's pooling rules and save the results in an object called `estimates`. Lines 24 and 25 print the pooled estimates and confidence intervals.

Following a significant group-by-time interaction effect, researchers typically examine the slope of the focal predictor at different values of the moderator. The final code block below computes these conditional effects or simple slopes of the monthly change rate at each value of `CONDITION`. The `constraints` parameter is a text string that defines the computation of the conditional growth rate in each subgroup.

R Script Ex18.R, continued

```
26  testConstraints(fit, constraints = 'month7 + month7*condition*0')
27  testConstraints(fit, constraints = 'month7 + month7*condition*1')
```

18.6 R Output

The table of unstandardized pooled parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third through fifth columns display the corresponding test statistics. The focal model results are shown below. The `RIV` column (relative increase in variance) is a fraction comparing imputation noise to complete-data sampling variation, and the `FMI` column (fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
Final parameter estimates and inferences obtained from 20 imputed data sets.

                 Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
(Intercept)        52.872     0.745    70.960 2.156e+04     0.000     0.031     0.030
month7              0.453     0.090     5.050 2.055e+03     0.000     0.106     0.097
male.cgm            0.312     0.225     1.389 4.971e+04     0.165     0.020     0.020
frlunch.cgm        -0.295     0.306    -0.966 2.469e+03     0.334     0.096     0.088
teachexp.cgm        0.000     0.001     0.144 1.986e+07     0.886     0.001     0.001
condition           1.559     0.992     1.571 6.291e+04     0.116     0.018     0.017
month7:condition    0.300     0.118     2.552 4.434e+03     0.011     0.070     0.066

                                  Estimate
Intercept~~Intercept|student:school   11.012
month7~~month7|student:school          0.043
```

```
Intercept~~month7|student:school       0.307
Intercept~~Intercept|school            6.305
month7~~month7|school                  0.075
Intercept~~month7|school               0.532
Residual~~Residual                    12.588

Unadjusted hypothesis test as appropriate in larger samples.

                       2.5 %       97.5 %
(Intercept)      51.41139113 54.332267087
month7            0.27726820  0.629304843
male.cgm         -0.12836247  0.752495046
frlunch.cgm      -0.89521085  0.304298130
teachexp.cgm     -0.00198809  0.002302489
condition        -0.38575795  3.503709185
month7:condition  0.06952040  0.530443946
```

The results are interpreted in the same way as a complete-data multilevel analysis. Lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of MONTH7 ($\hat{\gamma}_{10} = 0.45$) is the monthly change rate for students in the comparison condition (CONDITION = 0), and the intervention slope ($\hat{\gamma}_{04} = 1.56$) similarly reflects the mean difference when MONTH7 = 0 (at the final assessment). The interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\gamma}_{11} = 0.30$) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The corresponding test statistic indicates that the interaction effect is statistically different from zero ($t = 2.55$, $p = .01$). Note that these estimates are numerically equivalent to those from MCMC estimation. The output also includes pooled estimates of the variance–covariance parameters at all levels.

Finally, the printed output also includes the table of conditional effects or simple slopes. Consistent with the positive interaction coefficient, the monthly growth rate for the experimental schools is higher (more positive) that the growth rate for controls.

```
Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                                Estimate Std. Error
   month7 + month7*condition*0:    0.453      0.090
```

```
Combination method: D1

     F.value      df1      df2    P(>F)      RIV
      25.506        1 1332.100    0.000    0.106

Unadjusted hypothesis test as appropriate in larger samples.

Hypothesis test calculated from 20 imputed data sets. The following
constraints were specified:

                                 Estimate Std. Error
   month7 + month7*condition*1:      1.157      0.377

Combination method: D1

      F.value       df1       df2     P(>F)       RIV
        9.410         1 1.059e+05     0.002     0.011

Unadjusted hypothesis test as appropriate in larger samples.
```

18.7 Analyzing Multiple Imputations in Mplus

Multiple imputations for Mplus are created through the Blimp Studio interface. Returning to the previous Blimp script, the SAVE command and the separate keyword saved each imputed data set to a separate file with the asterisk replaced by a numeric index. The separate keyword also creates a list of file names needed for analysis in Mplus (in this example, a filed called implist.dat located in the imps subfolder). The contents of this file were shown in Section 12.6. The Mplus input file for analyzing the imputations is Ex18.inp.

Mplus Script Ex18.inp

```
1  DATA:
2  file = ./imps/implist.dat;
3  type = imputation;
4  VARIABLE:
5  names = school student wave condition teachexp eslpct ethnic
6    male frlunch lowach stanmath month0 month7 probsolve efficacy;
```

```
7   usevariables = probsolve month7 male frlunch teachexp condition;
8   cluster = school student;
9   within = month7;
10  between = (student) male frlunch (school) teachexp condition;
11  DEFINE:
12  center male frlunch teachexp (grandmean);
13  ANALYSIS:
14  type = threelevel random;
15  MODEL:
16  %within%
17  ranslope | probsolve on month7;
18  %between student%
19  probsolve on male frlunch;
20  probsolve with ranslope;
21  %between school%
22  [ranslope] (beta1);
23  probsolve on teachexp condition;
24  ranslope on condition (beta6);
25  ranslope with probsolve;
26  MODEL CONSTRAINT:
27  new(slp_cond0 slp_cond1);
28  slp_cond0 = beta1;
29  slp_cond1 = beta1 + beta6;
30  OUTPUT:
31  cinterval;
```

The DATA command lists the text file containing the names of the imputed data sets (the implist.dat file located in the ./imps subdirectory). The type = imputation subcommand instructs Mplus that the input data is a list of file names. The VARIABLE command provides information about the data. Beginning on line 5, the names subcommand assigns names to the variables in the input data file, and the usevariables subcommand selects variables for the analysis. The cluster subcommand on line 8 lists the school- and student-level identifier variables that indicate the clustering of the data records. The within and between subcommands on lines 9 and 10 identify level-1, level-2, and level-3 predictors.

On line 12, the center subcommand under the DEFINE command centers the three covariates at their grand means. The ANALYSIS command and type = threelevel random subcommand is required for estimating three-level models with random slopes at each level. The MODEL section of the script consists of three sections: the %within% section specifies the regression of the outcome

on level-1 time scores, and the `%between student%` section specifies the regression of the random intercepts on the level-2 predictors, and the `%between school%` section specifies the regression of the random intercepts on the level-3 predictors. In the `%within%` section, listing `ranslope` (an arbitrary name) to the left of the vertical pipe creates level-2 and level-3 latent variable capturing growth rates. Regressing this latent variable on `CONDITION` in the `%between school%` model gives the cross-level interaction. Beginning on line 26, the `MODEL CONSTRAINT` command is used to compute conditional effects or simple slopes. First, line 27 assigns names to two new parameters (the group-specific growth rates). Lines 28 and 29 use parameter labels from the `MODEL` section to compute the conditional effect of `MONTH7` in each experimental condition.

18.8 Mplus Output

The table of unstandardized parameter estimates is shown below. The first two columns display the pooled unstandardized estimates and standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown below. The `Rate of Missing` column (also called the fraction of missing information in the literature) quantifies the imputation noise in each estimate as proportion of its squared standard error.

```
MODEL RESULTS

                                                    Two-Tailed   Rate of
                  Estimate       S.E.  Est./S.E.    P-Value    Missing
Within Level

 Residual Variances
    PROBSOLVE        12.563      0.812     15.467      0.000      0.013

 Between STUDENT Level

  PROBSOLVE  ON
    MALE              0.336      0.258      1.301      0.193      0.017
    FRLUNCH          -0.304      0.308     -0.986      0.324      0.084

  PROBSOLV WITH
    RANSLOPE          0.300      0.144      2.084      0.037      0.101

  Variances
    RANSLOPE          0.042      0.028      1.464      0.143      0.143
```

```
 Residual Variances
    PROBSOLVE         10.970      0.827     13.272      0.000      0.098

Between SCHOOL Level

 RANSLOPE   ON
    CONDITION          0.301      0.113      2.663      0.008      0.058


PROBSOLVE  ON
    TEACHEXP           0.014      0.073      0.191      0.848      0.033
    CONDITION          1.558      0.949      1.642      0.101      0.015

 RANSLOPE WITH
    PROBSOLVE          0.490      0.180      2.717      0.007      0.052

 Intercepts
    PROBSOLVE         52.855      0.735     71.869      0.000      0.020
    RANSLOPE           0.449      0.080      5.588      0.000      0.083

 Residual Variances
    PROBSOLVE          5.696      1.905      2.989      0.003      0.026
    RANSLOPE           0.070      0.021      3.336      0.001      0.063

New/Additional Parameters
    SLP_COND           0.449      0.080      5.588      0.000      0.083
    SLP_COND           0.749      0.079      9.545      0.000      0.017
```

Mplus separates level-specific effects on the output (labeled `Within Level` and `Between STUDENT Level`, and `Between SCHOOL Level`). Considering the coefficients, lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of `MONTH7` ($\hat{\beta}_1 = 0.45$) is the monthly change rate for students in the comparison condition (`CONDITION` = 0), and the intervention slope ($\hat{\beta}_5 = 1.56$) similarly reflects the mean difference when `MONTH7` = 0 (at the final assessment). The interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\beta}_6 = 0.30$) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The corresponding test statistic indicates that the interaction is

statistically different from zero (z = 2.66, p = .01). Finally, the printed output also includes the table of conditional effects, which were computed using the `MODEL CONSTRAINT` command. Consistent with the positive interaction coefficient, the simple slope for the experimental schools is higher (more positive) that the growth rate for controls. Note that these estimates are numerically equivalent to those from MCMC estimation.

18.9 Analyzing Multiple Imputations in SPSS

Multiple imputations for SPSS and other commercial software packages are obtained through the Blimp Studio interface. Returning to the previous Blimp script, the `SAVE` command and the `stacked` keyword saved the imputed data sets to a single stacked file with an index variable in the first column identifying the individual files. The SPSS workbook file for the analysis is `Ex18.spwb`. The code block below shows the commands that import the stacked text file produced by Blimp. The example assumes that the data file is located on the desktop.

SPSS Script Ex18.spwb

```
CD '/users/username/desktop'.
DATA LIST free file =  'imps.dat'
  /imputation_ school student wave condition teachexp eslpct ethnic male
   frlunch lowach stanmath month0 month7 probsolve efficacy.
EXE.

AGGREGATE
  /OUTFILE = * MODE = ADDVARIABLES OVERWRITE = YES
  /male_mean = MEAN(male)
  /frlunch_mean = MEAN(frlunch)
  /teachexp_mean = MEAN(teachexp).

COMPUTE male_cgm = male - male_mean.
COMPUTE frlunch_cgm = frlunch - frlunch_mean.
COMPUTE teachexp_cgm = teachexp - teachexp_mean.
EXE.
```

The first line uses the `CD` command to change the working directory to the desktop. `username` portion of the file path should be replaced with the user's own account name. The data command uses a relative file path to read the stacked data file from the desktop. Variable names are listed

beginning on line 3. Importantly, the first variable named IMPUTATION_ is the index that identifies the individual files. SPSS reserves this exact variable name for multiply imputed data, and the pooling routines will not function if the index variable has a different name. On line 6, the AGGREGATE command adds the grand means to the data. Then, beginning on line 13, covariates are centered at their pooled grand means.

The next block of code fits the model to each data set and pools the results using Rubin's rules. The SORT command on line 17 sorts the data by the imputation index variable, and the SPLIT FILE command on line 18 triggers Rubin's pooling rules for all analyses that follow. The analysis syntax, which can be pasted from the pull-down menus, begins on line 19.

SPSS Script Ex18.spwb, continued

```
17  SORT CASES by imputation_.
18  SPLIT FILE layered by imputation_.
19  MIXED probsolve with month7 male_cgm frlunch_cgm teachexp_cgm condition
20    /print = solution testcov
21    /fixed = month7 male_cgm frlunch_cgm teachexp_cgm
22     condition month7*condition
23    /random = intercept month7 | subject(school) covtype(un)
24    /random = intercept month7 | subject(school*student) covtype(un).
```

18.10 SPSS Output

SPSS offers very little customization. Not every estimate on the output is pooled, and significance tests are generally limited to univariate *t* tests of individual parameters. Output tables display the analysis results for each data set, and the pooled results are at the bottom of each table (if they are produced). The figure below shows the pooled coefficients, standard errors, and test statistics. The output also includes pooled estimates of level-2 variance and covariance parameters. The relative increase in variance is a fraction comparing imputation noise to complete-data sampling variation, and the fraction of missing information quantifies the imputation noise in each estimate as proportion of its squared standard error.

The pooled regression coefficients are in the table labeled Estimates of Fixed Effects. The results are interpreted in the same way as a complete-data multilevel analysis. Lower-order terms in a moderated regression are conditional effects that depend on scaling or centering. Specifically, the lower-order slope of MONTH7 ($\hat{\gamma}_{10} = 0.46$) is the monthly change rate for students

in the comparison condition (`CONDITION` = 0), and the intervention slope ($\hat{\gamma}_{04}$ = 1.55) similarly reflects the mean difference when `MONTH7` = 0 (at the final assessment). The interaction effect captures the growth rate difference for students in experimental schools. The positive coefficient ($\hat{\gamma}_{11}$ = 0.30) indicates that the growth rate for the experimental condition is greater (more positive) than that of the comparison condition. The corresponding test statistic indicates that the interaction effect is statistically different from zero (t = 2.55, p = .011)[1]. The pooled random intercept and within-cluster residual variances are in the table labeled `Estimates of Covariance Parameters`.

Estimates of Fixed Effects[a]

imputation_	Parameter	Estimate	Std. Error	df	t	Sig.	95% Confidence Interval		Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
							Lower Bound	Upper Bound			
1.00	Intercept	52.812656	.712383	25.797	74.135	.000	51.347770	54.277541			
	month7	.436529	.085820	28.126	5.087	.000	.260771	.612288			
	malecent	.309541	.221891	964.407	1.395	.163	−.125904	.744985			
	frlunchcent	−.271113	.291839	971.145	−.929	.353	−.843820	.301594			
	teachexpcent	.039813	.076504	27.766	.520	.607	−.116958	.196583			
	condition	1.575874	.957060	25.504	1.647	.112	−.393254	3.545002			
	month7 * condition	.313885	.114327	27.016	2.745	.011	.079311	.548459			
2.00	Intercept	52.866443	.706710	27.005	74.806	.000	51.416407	54.316480			
	month7	.436348	.079615	28.574	5.481	.000	.273411	.599285			
	malecent	.275508	.223543	964.570	1.232	.218	−.163178	.714195			
	frlunchcent	−.253860	.294999	974.022	−.861	.390	−.832766	.325046			
	teachexpcent	.015963	.080736	27.748	.198	.845	−.149484	.181411			
	condition	1.520443	.945698	26.412	1.608	.120	−.421995	3.462880			
	month7 * condition	.305280	.105818	27.222	2.885	.008	.088241	.522319			
...											
20.00	Intercept	52.665831	.705523	26.374	74.648	.000	51.216609	54.115053			
	month7	.410935	.086035	28.091	4.776	.000	.234727	.587144			
	malecent	.294414	.221475	964.247	1.329	.184	−.140215	.729042			
	frlunchcent	−.257602	.293433	972.273	−.878	.380	−.833436	.318232			
	teachexpcent	.016738	.082533	27.695	.203	.841	−.152406	.185883			
	condition	1.790814	.945406	25.902	1.894	.069	−.152854	3.734483			
	month7 * condition	.349147	.114618	26.986	3.046	.005	.113965	.584329			
Pooled	Intercept	52.850066	.739177		71.499	.000	51.401269	54.298862	.020	.020	.999
	month7	.449336	.088951		5.052	.000	.274945	.623728	.068	.073	.997
	malecent	.335567	.225113		1.491	.136	−.105660	.776795	.023	.023	.999
	frlunchcent	−.296345	.305982		−.969	.333	−.896332	.303642	.085	.093	.996
	teachexpcent	.013961	.079731		.175	.861	−.142317	.170238	.028	.029	.999
	condition	1.557644	.987679		1.577	.115	−.378195	3.493483	.014	.014	.999
	month7 * condition	.299239	.117588		2.545	.011	.068729	.529750	.054	.057	.997

a. Dependent Variable: probsolve.

[1] For unknown reasons, the SPSS results differ from the R and Mplus imputation results. This is presumably due to differences in optimizers.

Estimates of Covariance Parameters[a]

nputation_	Parameter		Estimate	Std. Error	Wald Z	Sig.	95% Confidence Interval		Fraction Missing Info.	Relative Increase Variance	Relative Efficiency
							Lower Bound	Upper Bound			
.00	Residual		12.634262	.254991	49.548	.000	12.144246	13.144051			
	Intercept + month7 [subject = school]	UN (1,1)	5.926969	1.833908	3.232	.001	3.231886	10.869494			
		UN (2,1)	.504023	.188401	2.675	.007	.134763	.873283			
		UN (2,2)	.076779	.025370	3.026	.002	.040177	.146727			
	Intercept + month7 [subject = school * student]	UN (1,1)	10.612257	.764462	13.882	.000	9.214904	12.221506			
		UN (2,1)	.237927	.108446	2.194	.028	.025377	.450476			
		UN (2,2)	.031698	.023920	1.325	.185	.007223	.139115			
.00	Residual		12.540315	.253095	49.548	.000	12.053942	13.046313			
	Intercept + month7 [subject = school]	UN (1,1)	5.786742	1.760271	3.287	.001	3.187916	10.504160			
		UN (2,1)	.449483	.169397	2.653	.008	.117472	.781494			
		UN (2,2)	.062542	.021597	2.896	.004	.031786	.123057			
	Intercept + month7 [subject = school * student]	UN (1,1)	11.461198	.801028	14.308	.000	9.993997	13.143796			
		UN (2,1)	.389335	.114642	3.396	.001	.164641	.614029			
		UN (2,2)	.062158	.025041	2.482	.013	.028221	.136904			
• •											
0.00	Residual		12.764188	.257613	49.548	.000	12.269132	13.279219			
	Intercept + month7 [subject = school]	UN (1,1)	5.805647	1.778912	3.264	.001	3.184443	10.584440			
		UN (2,1)	.491725	.184857	2.660	.008	.129412	.854039			
		UN (2,2)	.077255	.025515	3.028	.002	.040439	.147588			
	Intercept + month7 [subject = school * student]	UN (1,1)	10.577117	.765791	13.812	.000	9.177824	12.189753			
		UN (2,1)	.234596	.108686	2.158	.031	.021574	.447617			
		UN (2,2)	.026512	.023933	1.108	.268	.004519	.155542			
ooled	Residual		12.562734	.269812		.000	12.033450	13.092017	.118	.132	.994
	Intercept + month7 [subject = school]	UN (1,1)	6.275891	1.946745		.001	2.460129	10.091652	.030	.030	.999
		UN (2,1)	.531770	.199219		.008	.141246	.922293	.050	.052	.998
		UN (2,2)	.076647	.026100		.003	.025485	.127809	.048	.050	.998
	Intercept + month7 [subject = school * student]	UN (1,1)	10.995674	.821400		.000	9.384785	12.606563	.098	.108	.995
		UN (2,1)	.299977	.119521		.012	.065416	.534537	.144	.166	.993
		UN (2,2)	.041728	.026462		.115	-.010225	.093681	.165	.194	.992

a. Dependent Variable: probsolve.

19

FIML and MCMC: Selection Model for Regression

This example illustrates a multiple regression analysis with a selection model that invokes a missing not at random process for the outcome. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric
	Missing Data Indicator		
M	9th grade reading missingness indicator	0	0 = observed, 1 = missing

19.1 Analysis Model

The analysis model features ninth grade broad reading scores regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1^*) + \varepsilon \tag{33}$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following the same factored regression specification from earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

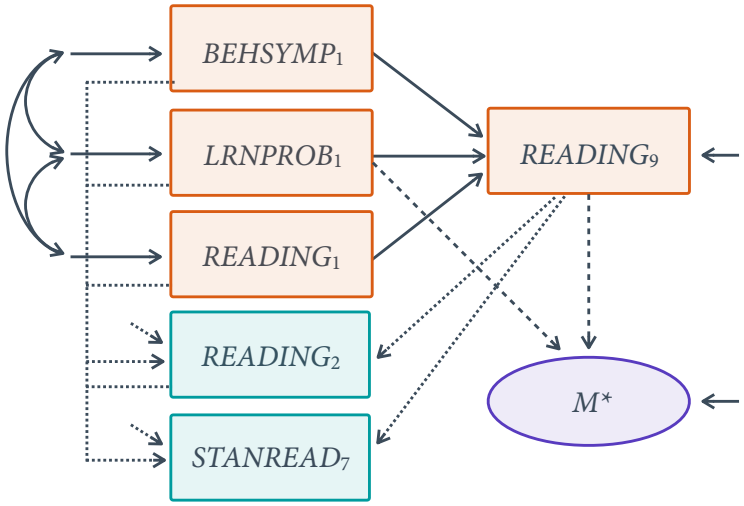
$$\begin{aligned} READ_2 = \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \ \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1^*) + \epsilon_1 \\ STANREAD_7 = \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \ \gamma_{32}(READ_1) \\ + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1^*) + \epsilon_2 \end{aligned} \tag{34}$$

Along with the focal regression model from Equation 34, the collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network (see the path diagram from Section 2.4). The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory.

A missing not at random process is invoked by specifying a selection model that links the missingness probabilities to the unseen outcome scores. This model features the binary missing data indicator regressed on the outcome variable and potentially other variables. To avoid excessive overlap between the focal and missingness models, the selection model used first grade learning problems as an additional regressor.

$$M^* = \gamma_{03} + \gamma_{13}(READ_9) + \gamma_{23}(LRNPROB_1) + \epsilon_3 \tag{35}$$

The asterisk superscript denotes a normally distributed latent response variable (i.e., a probit regression). A path diagram of the focal and selection models is shown below, with dashed lines



indicator the missingness model parameters. The oval labeled M^* represents latent response variable for the missingness indicator.

19.2 Mplus FIML Script

The code block below shows Mplus script `Ex19.inp`.

Mplus Script Ex19.inp

```
DATA:
file = behaviorachievement.dat;
VARIABLE:
names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
  read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7;
usevariables = read9 read1 lrnprob1 behsymp1 read2 stanread7 m;
missing = all(999);
categorical = m;
DATA MISSING:
names = read9;
binary = m;
type = missing;
```

```
14  ANALYSIS:
15  estimator = ml;
16  link = probit;
17  integration = montecarlo;
18  MODEL:
19  read1 lrnprob1 behsymp1;
20  read9 on read1 lrnprob1 behsymp1;
21  m on read9 lrnprob1;
22  read2 on read9 read1 lrnprob1 behsymp1;
23  stanread7 on read2 read9 read1 lrnprob1 behsymp1;
24  OUTPUT:
25  patterns sampstat stdyx cinterval;
```

The DATA command specifies the name of the input text file. No file path is required when the data set is in the same directory as the script, as it is here. The VARIABLE command provides information about the data. Beginning on line 4, the names subcommand assigns names to the variables in the input data, the usevariables subcommand selects variables for the analysis, and the missing subcommand gives the global missing value code. Lines 10 through 13 define a binary missing data indicator called M, and the preceding categorical subcommand on line 9 identifies the new variable as categorical.

The DATA MISSING command that begins on line 10 creates a binary missing data indicator. The names subcommand on line 11 identifies the variable to be recoded, and the binary command on line 12 provides a name for the new variable. Finally, the type subcommand on line 13 identifies the binary variable as a missing data indicator. As noted previously, the missingness indicator is identified as a categorical variable on line 9.

The ANALYSIS command and estimator subcommand specify full information maximum likelihood estimation. The default setting for a binary outcome is logistic regression. For consistency with the MCMC analysis in Blimp, line 16 specifies a probit link that defines the binary missing data indicator as a normally distributed latent response variable. Finally, the integration = montecarlo subcommand invokes an algorithmic method for models with mixed variable types.

The MODEL section of the script consists of five lines. Listing all predictors by name on line 19 is important because doing so invokes a multivariate normal distribution for these variables. As mentioned previously, assigning distributional assumptions to predictors is necessary for missing data handling. On line 20, the outcome variable appears to the left of the on keyword, and the

predictors appear to the right. The missingness model from Equation 36 appears on line 21, and the two auxiliary variable regressions from Equation 35 are on lines 22 and 23. Finally, the OUTPUT command specifies four keywords on line 25 that request a summary of the missing data patterns, maximum likelihood estimates of sample statistics, standardized coefficients, and confidence intervals.

19.3 Mplus Output

Information about the missing data patterns is found near the top of the output file. Following the missing data pattern table, the output displays a covariance coverage matrix that gives the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. The format of these tables is the same as those shown in Section 1.3.

The table of unstandardized parameter estimates is shown below. Because the analysis specifies a multivariate normal distribution for the predictors, the means, variances, and covariances of these variables are printed along with the focal model estimates. The table also reports regression models for auxiliary variables. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values.

```
MODEL RESULTS

                                                    Two-Tailed
                  Estimate       S.E.  Est./S.E.    P-Value

 READ9      ON
    READ1           0.507      0.042     12.201      0.000
    LRNPROB1       -0.251      0.116     -2.170      0.030
    BEHSYMP1       -0.180      0.101     -1.783      0.075

M          ON
    READ9          -0.006      0.010     -0.633      0.527
    LRNPROB1        0.042      0.013      3.150      0.002

READ2      ON
    READ9           0.676      0.065     10.373      0.000
```

```
    READ1              0.548      0.044     12.474      0.000
    LRNPROB1          -0.284      0.083     -3.428      0.001
    BEHSYMP1           0.412      0.076      5.395      0.000

 STANREAD7  ON
    READ2              1.903      0.924      2.060      0.039
    READ9              1.559      0.842      1.852      0.064
    READ1             -0.736      0.608     -1.210      0.226
    LRNPROB1           0.540      0.662      0.817      0.414
    BEHSYMP1          -0.753      0.658     -1.144      0.253

 LRNPROB1 WITH
    READ1            -11.635     19.119     -0.609      0.543

 BEHSYMP1 WITH
    READ1            -14.114     21.254     -0.664      0.507
    LRNPROB1          91.527     13.505      6.777      0.000

 Means
    READ1             86.154      1.752     49.188      0.000
    LRNPROB1          52.292      0.915     57.121      0.000
    BEHSYMP1          49.483      1.034     47.851      0.000

Intercepts
    READ9             65.832      5.832     11.287      0.000
    READ2            -19.011      5.741     -3.311      0.001
    STANREAD7         19.329     50.325      0.384      0.701

 Thresholds
    M$1          2.715      1.305      2.080      0.038

Variances
    READ1            417.284     51.743      8.065      0.000
    LRNPROB1         114.548     13.883      8.251      0.000
    BEHSYMP1         145.486     17.587      8.272      0.000

Residual Variances
    READ9             86.368     11.474      7.528      0.000
    READ2             38.774      5.663      6.847      0.000
    STANREAD7       2206.056    303.868      7.260      0.000
```

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 12.20$, $p < .001$). Comparing these results to the estimates that invoke a conditionally missing at random process provides a sensitivity check (see Section 1.3). Because the selection model estimates are virtually identical to those from Chapter 1, one can conclude that the regression parameters are somewhat robust to a different missingness process. This interpretation presupposes that the missingness model is correctly specified. A different set of predictors in the selection equation could change the estimates and the conclusion about robustness.

The table also reports the missingness model parameters. The outcome variable is a latent response score that represents a normally distributed propensity for missingness. To establish a metric, the latent responses are approximately scaled as a *z*-score. Thus, the missingness model slope coefficients essentially represent the standardized change in the missingness propensities for a one-unit increase in the predictors. The negative coefficient for READ9 suggests that students with higher ninth grade reading scores have a lower probability of missing data in ninth grade, and the positive slope for LRNPROB1 indicates that students with elevated learning problems in first grade are more likely to have missing data in middle school.

19.4 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script Ex19.inp. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex19.imp

```
DATA: behaviorachievement.dat;
VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
   read1 read2 read3 read9 read9grp stanread7
   math1 math2 math3 math9 math9grp stanmath7;
MISSING: 999;
TRANSFORM:
m = ismissing(read9);
ORDINAL: m;
MODEL:
focal model:
```

```
11  read9 ~ read1 lrnprob1 behsymp1;
12  missingness.model:
13  m ~ read9 lrnprob1;
14  auxiliary model:
15  stanread7 read2 ~ read9 read1 lrnprob1 behsymp1;
16  SEED: 90291;
17  BURN: 1000;
18  ITERATIONS: 10000;
```

The first eight lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns, and the MISSING command on line 5 defines a global missing value code as 999. The TRANSFORM command that starts on line 6 uses the ismissing function to create a binary missing data indicator called M. The ORDINAL command on line 8 identifies the indicator as a binary variable.

The MODEL command that begins on line 9 lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The focal model is listed on line 11, and the missingness (selection) model is on line 13. Line 15 is a syntax shortcut that produces the two auxiliary variable regression models in Equation 35; in the first model, READ2 is regressed on the focal variables, and the second model features STANREAD7 regressed on READ2 and the focal variables. Finally, note that the MODEL block uses labels to order the regression summary tables on the output.

Lines 16 through 18 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex19.R is shown below.

rblimp Script Ex19.R

```
1  library(rblimp)
2  load('behaviorachievement.rda')
```

```
3
4  mymodel <- rblimp(
5    data = behaviorachievement,
6    transform = 'm = ismissing(read9)',
7    ordinal = 'm',
8    model = '
9      focal.model:
10     read9 ~ read1 lrnprob1 behsymp1;
11     missingness.model:
12     m ~ read9 lrnprob1;
13     auxiliary.model:
14     stanread7 read2  ~ read9 read1 lrnprob1 behsymp1',
15   seed = 90291,
16   burn = 1000,
17   iter = 10000)
18 output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the `rblimp` function. The two exceptions are the `VARIABLES` and `MISSING` commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the `MODEL` command) are separated by semicolons, as they are in the Blimp script.

19.5 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           26 to 50              1.273            23
                           51 to 100             1.074            40
                           76 to 150             1.081            12
                                 ...               ...            ..
                          451 to 900             1.011            14
                          476 to 950             1.007            12
                          501 to 1000            1.015            17
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  read9

Parameters                      Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
Variances:
  Residual Var.                 91.783     12.981     70.696    121.147        ---        ---   6177.312

Coefficients:
  Intercept                     66.409      5.993     54.281     77.590    122.234      0.000   6416.982
  read1                          0.505      0.043      0.422      0.591    136.863      0.000   6710.053
  lrnprob1                      -0.254      0.119     -0.486     -0.017      4.533      0.033   5463.375
  behsymp1                      -0.183      0.103     -0.386      0.016      3.167      0.075   6391.699

Standardized Coefficients:
  read1                          0.687      0.040      0.599      0.755    297.770      0.000   6461.097
  lrnprob1                      -0.181      0.084     -0.341     -0.012      4.653      0.031   5414.690
  behsymp1                      -0.147      0.082     -0.306      0.013      3.220      0.073   6405.638

Proportion Variance Explained
  by Coefficients                0.596      0.050      0.487      0.681        ---        ---   6280.850
  by Residual Variation          0.404      0.050      0.319      0.513        ---        ---   6280.850

-------------------------------------------------------------------------------------------------------
```

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue. Unlike previous examples, this analysis specified 20,000 iterations because the effective sample size for the random slope variance was less than 100 when using 10,000 iterations.

The results are interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. The frequentist test statistic and p-value give the same conclusion. Comparing these results to estimates that invoke a conditionally missing at random process provides a sensitivity check (see Section 6.3). Because

the selection model estimates are virtually identical to those from Chapter 6, one can conclude that the regression parameters are somewhat robust to a different missingness process. This interpretation presupposes that the missingness model is correctly specified. A different set of predictors in the selection equation could change the estimates and the conclusion about robustness.

The table also reports the missingness model parameters. The outcome variable is a latent response score that represents a normally distributed propensity for missingness. To establish a metric, the latent responses are approximately scaled as a *z*-score. Thus, the missingness model slope coefficients essentially represent the standardized change in the missingness propensities for a one-unit increase in the predictors. The negative coefficient for READ9 suggests that students with higher ninth grade reading scores have a lower probability of missing data in ninth grade, and the positive slope for LRNPROB1 indicates that students with elevated learning problems in first grade are more likely to have missing data in middle school. Note that an unusually large *R*-squared value in the missingness model (e.g., greater than 70%) is often a symptom of overfitting the selection equation with too many predictors. This analysis does not exhibit that symptom.

```
  missingness.model block:

Outcome Variable:  m

Parameters                     Median     StdDev       2.5%      97.5%      ChiSq     pvalue      N_Eff
-------------------------------------------------------------------------------------------------------
Variances:
  Residual Var.                 1.000      0.000      1.000      1.000        ---        ---        nan

Coefficients:
  Intercept                    -2.084      1.133     -4.296      0.160      3.379      0.066   2687.691
  read9                        -0.011      0.009     -0.029      0.007      1.360      0.243   2256.809
  lrnprob1                      0.038      0.012      0.014      0.062      9.672      0.002   2739.452

Thresholds:
  Tau 1                         0.000      0.000      0.000      0.000        ---        ---        nan

Standardized Coefficients:
  read9                        -0.143      0.119     -0.370      0.099      1.406      0.236   2323.761
  lrnprob1                      0.361      0.102      0.144      0.543     12.337      0.000   2833.886

Proportion Variance Explained
  by Coefficients               0.198      0.081      0.058      0.370        ---        ---   2190.883
  by Residual Variation         0.802      0.081      0.630      0.942        ---        ---   2190.883

-------------------------------------------------------------------------------------------------------
```

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

20

FIML and MCMC: Pattern Mixture Model For Regression

This example illustrates a pattern mixture regression model that invokes a missing not at random process for the outcome. The analysis uses the `behaviorachievement.dat` data set taken from a longitudinal study that followed 138 students from primary through middle school. The file includes three annual assessments of broad reading and math achievement beginning in the first grade, seventh grade standardized achievement test scores taken from a statewide assessment, and a final measure of broad reading and math obtained in ninth grade. The data also contain teacher ratings of behavioral symptoms and learning problems were also obtained in the first grade. The data description at the beginning of this document provides additional details. The variables for this analysis are as follows.

Name	Definition	Missing %	Scale
	Focal Variables		
$BEHSYMP_1$	1st grade behavioral symptoms	3.6	Numeric
$LRNPROB_1$	1st grade learning problems	2.2	Numeric
$READ_1$	1st grade broad reading composite	6.5	Numeric
$READ_9$	9th grade broad reading composite	17.4	Numeric
	Auxiliary Variables		
$READ_2$	2nd grade broad reading composite	9.4	Numeric
$STANREAD_7$	7th grade standardized math	19.6	Numeric
	Missing Data Indicator		
M	9th grade reading missingness indicator	0	0 = observed, 1 = missing

20.1 Analysis Model

The population-level analysis model features ninth grade broad reading scores regressed on first grade reading achievement and teacher-rated learning problems and behavioral symptoms.

$$READ_9 = \beta_0 + \beta_1(READ_1) + \beta_2(LRNPROB_1) + \beta_3(BEHSYMP_1^*) + \varepsilon \tag{36}$$

Unlike a complete-data regression analysis, all incomplete variables require distributional assumptions, including the predictors.

A missing not at random process is invoked by specifying a pattern mixture model that links the missingness probabilities to the unseen outcome scores. This model features the binary missing data indicator as a predictor and possibly a moderator. The basic idea is that the missing data patterns define subgroups with different parameter values. This example illustrates a process where students with missing scores in ninth grade have a lower reading mean. It is also possible for the regression coefficients to differ by pattern (see Enders, 2022, Section 9.8).

To invoke a missing data pattern-specific mean difference, the fitted model includes the binary missing data indicator as a predictor

$$\begin{aligned} READ_9 = &\left[\beta_{0(com)} + \beta_{0(mis)}(M)\right] + \beta_1(READ_1) \\ &+ \beta_2(LRNPROB_1) + \beta_3(BEHPROB_1) + \varepsilon \end{aligned} \tag{37}$$

such that $\beta_{0(\text{com})}$ is the intercept (mean level) for students with complete reading scores, and $\beta_{0(\text{diff})}$ is outcome mean difference for students with missing data.

Unlike a complete-data regression analysis, incomplete predictor variables also require distributional assumptions and models that define those distributions. The analysis uses a factored regression specification that uses a sequence of univariate regression models to link incomplete predictors. This specification was introduced throughout previous examples. The additional regression equations are as follows.

$$\begin{aligned} M^* &= \gamma_{01} + \epsilon_1 \\ BEHSYMP_1^* &= \gamma_{02} + \gamma_{12}(M) + \epsilon_2 \\ LRNPROB_1 &= \gamma_{03} + \gamma_{13}(BEHSYMP_1) + \gamma_{23}(M) + \epsilon_3 \end{aligned} \tag{38}$$

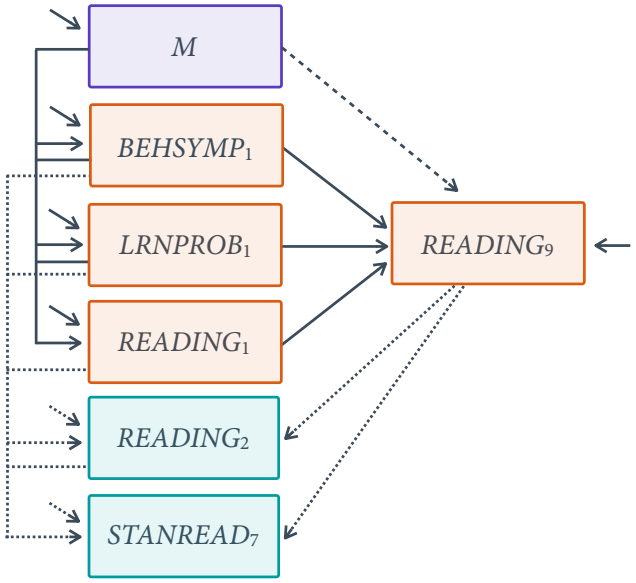
$$READ_1 = \gamma_{04} + \gamma_{14}(LRNPROB_1) + \gamma_{24}(BEHSYMP_1^*) + \gamma_{34}(M) + \epsilon_4$$

The asterisk subscript in the `M` model denotes a latent response variable (i.e., probit regression). Listing the missing data indicator first in the sequence is important because pattern proportions needed for Equation 39 are a function of the empty model's regression intercept.

The missing data literature often recommends an inclusive strategy that incorporates auxiliary variables that either predict missingness or correlate with the incomplete variables (Collins et al., 2001). Following the same factored regression specification from earlier examples, auxiliary variables enter the model as additional outcomes that are predicted by the analysis variables and by each other. The additional regression equations are as follows.

$$\begin{aligned} READ_2 &= \gamma_{01} + \gamma_{11}(READ_9) + \gamma_{21}(READ_1) + \gamma_{31}(LRNPROB_1) + \gamma_{41}(BEHSYMP_1^*) + \epsilon_1 \\ STANREAD_7 &= \gamma_{02} + \gamma_{12}(READ_2) + \gamma_{22}(READ_9) + \gamma_{32}(READ_1) \\ &\quad + \gamma_{42}(LRNPROB_1) + \gamma_{52}(BEHSYMP_1^*) + \epsilon_2 \end{aligned} \tag{39}$$

Along with the focal regression from Equation 37 and the predictor models from Equation 40, the collection of regressions can be viewed as a path model, where the focal regression is one part of a larger network. The key difference is that the path coefficients are just a tool for linking incomplete variables and do not represent a substantive theory. A path diagram of the full model is shown below.



The intercept coefficient from Equation 37 is a weighted average of the group-specific intercepts

$$\beta_0 = p_{(\text{com})}\beta_{0(\text{com})} + p_{(\text{mis})}\left(\beta_{0(\text{com})} + \beta_{0(\text{diff})}\right) = p_{(\text{com})}\beta_{0(\text{com})} + p_{(\text{mis})}\beta_{0(\text{mis})} \tag{40}$$

where $p_{(\text{com})}$ and $p_{(\text{mis})}$ are the proportions of complete and missing outcome scores, respectively. Importantly, $\beta_{0(\text{diff})}$ is not estimable from the data, and researchers must provide a value that induces the posited missing not at random process (e.g., students with missing outcome data have lower reading levels). Following the procedure described in Enders (2022), the scripts below set $\beta_{0(\text{diff})}$ to a value that is 0.20 standard deviation units below $\beta_{0(\text{com})}$. That is, the average reading level for students with missing outcome scores is lower by an amount commensurate with Cohen's (1988) small effect size benchmark.

20.2 Mplus FIML Script

The code block below shows Mplus script `Ex20.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Mplus Script Ex20.inp

```
DATA:
file = behaviorachievement.dat;
VARIABLE:
names = id male hispanic riskgrp atrisk behsymp1 lrnprob1
  read1 read2 read3 read9 read9grp stanread7
  math1 math2 math3 math9 math9grp stanmath7;
usevariables = read9 read1 lrnprob1 behsymp1 read2 stanread7 m;
missing = all(999);
categorical = m;
DATA MISSING:
names = read9;
binary = m;
type = missing;
ANALYSIS:
estimator = ml;
link = probit;
integration = montecarlo;
```

```
18  MODEL:
19  [m$1] (missmean);
20  behsymp1 on m;
21  lrnprob1 on behsymp1 m;
22  read1 on lrnprob1 behsymp1 m;
23  read9 on m (beta0diff)
24     read1 lrnprob1 behsymp1;
25  [read9] (beta0com); read9 (resvar);
26  read2 on read9 read1 lrnprob1 behsymp1 m;
27  stanread7 on read2 read9 read1 lrnprob1 behsymp1 m;
28  MODEL CONSTRAINT:
29  new(cohensd pcom pmis beta0);
30  cohensd = -.20;
31  beta0diff = cohensd * sqrt(resvar);
32  pmis = phi(-missmean);
33  pcom = 1 - pmis;
34  beta0 = (beta0com * pcom) + ((beta0com + beta0diff) * pmis);
35  OUTPUT:
36  patterns sampstat stdyx cinterval;
```

The DATA command specifies the name of the input text file. No file path is required when the data set is in the same directory as the script, as it is here. The VARIABLE command provides information about the data. Beginning on line 4, the names subcommand assigns names to the variables in the input data, the usevariables subcommand selects variables for the analysis, and the missing subcommand gives the global missing value code. Lines 10 through 13 define a binary missing data indicator called M, and the preceding categorical subcommand on line 9 identifies the new variable as categorical.

The DATA MISSING command that begins on line 10 creates a binary missing data indicator. The names subcommand on line 11 identifies the variable to be recoded, and the binary command on line 12 provides a name for the new variable. Finally, the type subcommand on line 13 identifies the binary variable as a missing data indicator. As noted previously, the missingness indicator is identified as a categorical variable on line 9.

The ANALYSIS command and estimator subcommand specify full information maximum likelihood estimation. The default setting for a binary outcome is logistic regression. For consistency with the MCMC analysis in Blimp, line 16 specifies a probit link that defines the binary missing data indicator as a normally distributed latent response variable. Finally, the

`integration = montecarlo` subcommand invokes an algorithmic method for models with mixed variable types.

The `MODEL` command that begins on line 18 lists the regression models, with outcome variables to the left of the `on` keyword and predictors to the right. An empty model for the missing data indicator is listed on line 19. The label on the threshold parameter from this model (`missmean`) is used later in the code to compute the missing data pattern proportions. The remaining predictor models from Equation 40 appear on lines 20 through 22. Next, lines 23 through 25 list the focal model parameters. Line 23 assigns a label to the pattern mean difference (i.e., the $\beta_{0(\text{diff})}$ coefficient from Equation 38), and line 25 labels the complete-case intercept and residual variance, respectively. Collectively, the labels are used later in the code to induce the desired effect size difference for the missing scores. Finally, lines 26 and 27 produce the two auxiliary variable regression models from Equation 41; in the first model, `READ2` is regressed on the focal variables, and the second model features `STANREAD7` regressed on `READ2` and the focal variables.

The `MODEL CONSTRAINT` section of the script from lines 28 through 34 includes commands that define new parameters and impose constraints. First, line 29 assigns names to four new parameters. Line 30 provides the desired effect size difference for the group with missing data, and line 31 defines a mean difference parameter `beta0diff` that is a function of the effect size and residual standard deviation (see Enders, 2022, Eq. 9.29). Lines 32 and 33 use the threshold parameter from the missing data indicator's model to compute the missing data pattern proportions. Line 34 computes the weighted intercept that averages over the missing data patterns (see Equation 39). Finally, the `OUTPUT` command specifies four keywords on line 36 that request a summary of the missing data patterns, maximum likelihood estimates of sample statistics, standardized coefficients, and confidence intervals.

20.3 Mplus Output

Information about the missing data patterns is found near the top of the output file. Following the missing data pattern table, the output displays a covariance coverage matrix that gives the proportion of observed data for each variable on the diagonal and the proportion of observed data for each variable pair on the off-diagonals. These tables are illustrated in Section 1.3.

The table of unstandardized parameter estimates is shown below. The table reports regression models for predictor variables and auxiliary variables. These supporting parameters are not of substantive interest, and they do not need to be reported. The first two columns display the unstandardized estimates and their standard errors, and the third and fourth columns display the corresponding *z*-statistics and *p*-values. The focal model results are shown in bold typeface.

MODEL RESULTS

	Estimate	S.E.	Est./S.E.	Two-Tailed P-Value
BEHSYMP1 ON				
M	7.692	2.654	2.898	0.004
LRNPROB1 ON				
BEHSYMP1	0.597	0.054	10.967	0.000
M	4.241	1.728	2.454	0.014
READ1 ON				
LRNPROB1	-0.012	0.244	-0.048	0.961
BEHSYMP1	-0.064	0.208	-0.306	0.760
M	-3.230	4.861	-0.664	0.506
READ9 ON				
M	-1.862	0.124	-14.959	0.000
READ1	**0.504**	**0.042**	**11.990**	**0.000**
LRNPROB1	**-0.248**	**0.117**	**-2.108**	**0.035**
BEHSYMP1	**-0.181**	**0.101**	**-1.790**	**0.073**
READ2 ON				
READ9	0.674	0.065	10.344	0.000
READ1	0.551	0.044	12.563	0.000
LRNPROB1	-0.290	0.084	-3.440	0.001
BEHSYMP1	0.414	0.077	5.415	0.000
M	1.124	2.244	0.501	0.617
STANREAD7 ON				
READ2	1.891	0.923	2.048	0.041
READ9	1.590	0.841	1.891	0.059
READ1	-0.733	0.609	-1.205	0.228
LRNPROB1	0.493	0.678	0.728	0.467
BEHSYMP1	-0.737	0.659	-1.118	0.264
M	6.119	13.186	0.464	0.643
Intercepts				

```
    READ9              66.040      5.887     11.218      0.000
    READ1              90.485      9.209      9.826      0.000
    LRNPROB1           21.999      2.715      8.103      0.000
    BEHSYMP1           48.148      1.104     43.610      0.000
    READ2             -19.003      5.776     -3.290      0.001
    STANREAD7          18.400     50.557      0.364      0.716

 Thresholds
    M$1         0.939      0.126      7.472      0.000

 Residual Variances
    READ9              86.643     11.584      7.480      0.000
    READ1             414.570     51.322      8.078      0.000
    LRNPROB1           54.545      6.727      8.109      0.000
    BEHSYMP1          137.009     16.565      8.271      0.000
    READ2              38.759      5.658      6.850      0.000
    STANREAD7        2200.450    303.283      7.255      0.000

New/Additional Parameters
    COHENSD            -0.200      0.000  *********      0.000
    PCOM                0.826      0.032     25.608      0.000
    PMIS                0.174      0.032      5.389      0.000
    BETA0              65.716      5.887     11.162      0.000
```

The regression slopes interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.51 points on READ9. The corresponding test statistic indicates that the slope coefficient is statistically different from zero ($z = 11.99$, $p < .001$). The M coefficient from the same table is the pattern mean difference $\beta_{0(\text{diff})}$ (see Equation 38). The MODEL CONSTRAINT command defined a set of new model parameters, including weighted average intercept. The table summarizing the additional parameters is shown below. These quantities are found under the table labeled New/Additional Parameters. The weighted intercept coefficient that averages over the missing data patterns is labeled beta0.

Comparing these results to estimates that invoke a conditionally missing at random process (see Section 1.3) provides a sensitivity check that conveys the impact of a missing not at random process where students with missing data have lower mean reading levels in ninth grade. This

comparison presupposes that the missingness model is correctly specified. The missing data indicator could also moderate associations in the regression model, in which case the estimates and conclusions about robustness could change.

20.4 Blimp and rblimp MCMC Scripts

The code block below shows Blimp script `Ex20.inp`. This script is executed in the Blimp Studio graphical interface. The corresponding R script is shown later in this section.

Blimp Script Ex20.imp

```
DATA: behaviorachievement.dat;
VARIABLES: id male hispanic riskgrp atrisk behsymp1 lrnprob1
   read1 read2 read3 read9 read9grp stanread7
   math1 math2 math3 math9 math9grp stanmath7;
MISSING: 999;
TRANSFORM:
m = ismissing(read9);
ORDINAL:  m;
MODEL:
focal model:
read9 ~ 1@beta0com m@beta0diff read1 lrnprob1 behsymp1;
indicator.model:
m ~ 1@missmean;
predictor.model:
read1 lrnprob1 behsymp1 ~ m;
auxiliary model:
stanread7 read2 ~ read9 read1 lrnprob1 behsymp1;
PARAMETERS:
cohensd = -.20;
beta0diff = cohensd * sqrt(read9.totalvar);
pmis = phi(missmean);
pcom = 1 - pmis;
beta0 = (beta0com * pcom) + ((beta0com + beta0diff) * pmis);
SEED: 90291;
BURN: 1000;
ITERATIONS: 10000;
```

The first eight lines can be viewed as a set of commands that specify information about the data and variables. The DATA command specifies the name of the input text file. No file path is required when the data file is in the same directory as the script, as it is here. Starting on line 2, the VARIABLES command names the data columns, and the MISSING command on line 5 defines a global missing value code as 999. The TRANSFORM command that starts on line 6 uses the ismissing function to create a binary missing data indicator called M. The ORDINAL command on line 8 identifies the indicator as a binary variable.

The MODEL command that begins on line 9 lists the regression models, with outcome variables to the left of the tilde and predictors to the right. The code uses model block labels (focal.model, indicator.model, predictor.model, and auxiliary.model) to group the regressions and order output tables. The focal model listed on line 11 assigns labels to intercept and the pattern mean difference (i.e., the $\beta_{0(\text{com})}$ and $\beta_{0(\text{diff})}$ coefficients from Equation 38) using the @ symbol. The labels are used later in the code to induce the desired effect size difference for the missing scores. An empty model for the missing data indicator is listed on line 13. The label on the intercept parameter is used later in the code to compute the missing data pattern proportions. Line 15 is a syntax shortcut that produces the predictor regression models in Equation 40; in the first model, BEHSYMP1 is regressed on the binary missing data indicator M, the second model features LRNPROB1 regressed on BEHSYMP1 and the indicator, and the third regression features READ1 regressed on all other predictors. Line 17 is a similar syntax shortcut that produces the two auxiliary variable regression models in Equation 41; in the first model, READ2 is regressed on the focal variables, and the second model features STANREAD7 regressed on READ2 and the focal variables.

The PARAMETERS section of the script from lines 18 through 23 includes commands that define new parameters and impose constraints. Line 19 provides the desired effect size difference for the group with missing data, and line 20 defines a mean difference parameter beta0diff that is a function of the effect size and estimated standard deviation, which is obtained by appending .totalvar to the focal dependent variable's READ9 (see Enders, 2022, Eq. 9.29). Lines 21 and 22 use the intercept parameter from the missing data indicator's model to compute the missing data pattern proportions. Finally, line 23 computes the weighted intercept that averages over the missing data patterns (see Equation 41).

Lines 24 through 26 can be viewed as a block of commands that specify features of the MCMC algorithm: the SEED command gives an integer string that initializes the random number generator, the BURN command specifies the number of iterations for the warm-up or burn-in period, and the ITERATIONS command gives the number of MCMC iterations on which the

analysis summaries are based (essentially, the number of MCMC cycles following the warm-up period).

The corresponding rblimp script Ex20.R is shown below.

rblimp Script Ex20.R

```
library(rblimp)
load('behaviorachievement.rda')

mymodel <- rblimp(
  data = behaviorachievement,
  transform = 'm = 1 - ismissing(read9)',
  ordinal = 'm',
  model = '
    focal.model:
    read9 ~ 1@beta0com m@beta0diff read1 lrnprob1 behsymp1;
    missingness.model:
    m ~ 1@missmean;
    predictor.model:
    read1 lrnprob1 behsymp1 ~ m;
    auxiliary.model:
    stanread7 read2  ~ read9 m read1 lrnprob1 behsymp1',
  parameters = 'cohensd = -.20;
    beta0diff = cohensd * sqrt(read9.totalvar);
    pmis = phi(missmean);
    pcom = 1 - pmis;
    beta0 = (beta0com * pcom) + ((beta0com + beta0diff) * pmis)',
  seed = 90291,
  burn = 1000,
  iter = 10000)
output(mymodel)
```

Each command in the Blimp script (each capitalized word) is an input parameter in the rblimp function. The two exceptions are the VARIABLES and MISSING commands, which are omitted because that information is contained in the R data file. Following R convention, the input parameters are separated by commas. Alphanumeric inputs like model statements, variable lists, transformations, and new parameters are enclosed in quotes. Numeric inputs like the seed and

number of iterations do not require quotes. Finally, subcommands that are part of the same command (e.g., different equations in the MODEL command) are separated by semicolons, as they are in the Blimp script.

20.5 Blimp and rblimp Output

Prior to inspecting the parameter estimates, it is important to investigate the potential scale reduction (PSR) factor diagnostics (Gelman & Rubin, 1992) to determine whether MCMC has converged. Blimp divides the burn-in period into 20 equal segments, and it computes the PSR diagnostic for every parameter. The table located near the top of the output reports the highest (worst) PSR value across all parameters in every model. A common recommendation is that these values should be less than 1.05 or perhaps 1.10 (Asparouhov & Muthén, 2010a; Gelman et al., 2014). If the PSR in the bottom row of the table (the final check of the burn-in period) is above these cutoffs, then rerun the analysis with a longer burn-in period.

```
BURN-IN POTENTIAL SCALE REDUCTION (PSR) OUTPUT:

  NOTE: Split chain PSR is being used. This splits each chain's
        iterations to create twice as many chains.

  Comparing iterations across 2 chains     Highest PSR   Parameter #
                           26 to 50              1.468            58
                           51 to 100             1.107            43
                           76 to 150             1.175             4
                                 ...               ...            ..
                          451 to 900             1.008            53
                          476 to 950             1.009            67
                          501 to 1000            1.023            67
```

The MCMC summary tables include unstandardized coefficients, standardized slopes, and variance explained effect size estimates. MCMC estimation produces a distribution for each model parameter. The median and standard deviation columns describe the center and spread of the posterior distributions; although they make no reference to drawing repeated samples, they are analogous—and numerically equivalent in most cases—to frequentist point estimates and standard errors. The 95% credible intervals in the rightmost columns give a range that captures 95% of the parameter's distribution. These are akin to confidence intervals, but the intervals describe parameter distributions rather than characteristics of repeated samples. Although

MCMC estimation is grounded in the Bayesian statistical paradigm, one can also view posterior medians, standard deviations, and credible intervals as surrogates for frequentist point estimates, standard errors, and confidence intervals. Levy and McNeish (2023) describe this perspective as "computational frequentism". Essentially, the researcher wants to operate within the frequentist framework, but they use MCMC to solve a difficult estimation problem. Missing data analyses are a compelling use case for computational frequentism because optimal likelihood-based solutions are not always available or easy to use. To facilitate this perspective, the Blimp output also includes a chi-square statistic and *p*-value for each model parameter (the Bayesian Wald test; Asparouhov & Muthén, 2021). These Wald tests are like squared *z*–statistics from maximum likelihood estimation, but MCMC generates the point estimate and "standard error" for the test.

The table summarizing the focal regression model is shown below.

```
OUTCOME MODEL ESTIMATES:

  Summaries based on 10000 iterations using 2 chains.

  focal.model block:

Outcome Variable:  read9
```

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
Variances:							
Residual Var.	91.694	13.046	70.522	121.601	---	---	5903.081
Coefficients:							
Intercept	69.026	6.111	57.127	80.988	127.522	0.000	6333.447
m	-2.983	0.158	-3.313	-2.694	355.535	0.000	6636.674
read1	0.503	0.044	0.419	0.590	133.963	0.000	7164.000
lrnprob1	-0.244	0.122	-0.482	-0.004	4.013	0.045	4863.923
behsymp1	-0.183	0.105	-0.393	0.021	3.083	0.079	6107.174
Standardized Coefficients:							
m	-0.076	0.005	-0.087	-0.067	208.040	0.000	40000.000
read1	0.693	0.040	0.605	0.762	295.885	0.000	6829.250
lrnprob1	-0.176	0.087	-0.343	-0.003	4.086	0.043	4902.866
behsymp1	-0.149	0.084	-0.317	0.018	3.135	0.077	6112.053
Proportion Variance Explained							
by Coefficients	0.587	0.051	0.476	0.676	---	---	6342.070
by Residual Variation	0.413	0.051	0.324	0.524	---	---	6342.070

To begin, the N_Eff values in rightmost column of the table give the effective number of MCMC samples for each parameter. These quantities essentially represent the number of independent estimates on which the parameter summaries are based after removing autocorrelations from the MCMC process. Gelman et al. (2014, p. 287) recommend values greater than 100. All values in the example table exceed this recommended minimum. In cases where the N_Eff values are insufficient, increasing the value on the ITERATIONS command will remedy the issue. Unlike previous examples, this analysis specified 20,000 iterations because the effective sample size for the random slope variance was less than 100 when using 10,000 iterations.

The regression slopes interpreted in the same way as a complete-data regression analysis. For example, consider the first-grade reading score slope. The model predicts that two individuals who differ by one point on READ1 but are the same on LRNPROB1 and BEHSYMP1 should differ by 0.50 points on READ9. The 95% credible interval limits suggest this effect is statistically different from zero ($p < .05$) because the null value is well outside the interval. This table does not display the regression intercept. Rather, Intercept and M coefficients are the pattern-specific parameters, $\beta_{0(\text{com})}$ and $\beta_{0(\text{diff})}$ (see Equation 38).

The PARAMETERS command defined a set of new model parameters, including weighted average intercept. The table summarizing the additional parameters is shown below.

GENERATED PARAMETERS:

Summaries based on 10000 iterations using 2 chains.

Parameters	Median	StdDev	2.5%	97.5%	ChiSq	pvalue	N_Eff
cohensd	-0.200	0.000	-0.200	-0.200	inf	0.000	2.001
beta0diff	-2.983	0.158	-3.313	-2.694	355.489	0.000	6638.348
pmis	0.826	0.032	0.757	0.883	646.515	0.000	3155.285
pcom	0.174	0.032	0.117	0.243	29.305	0.000	3155.285
beta0	66.565	6.143	54.605	78.557	117.332	0.000	6364.816

The weighted intercept coefficient that averages over the missing data patterns is labeled beta0. Comparing these results to estimates that invoke a conditionally missing at random process (see Section 6.3) provides a sensitivity check that conveys the impact of a missing not at random process where students with missing data have lower mean reading levels in ninth grade. This comparison presupposes that the missingness model is correctly specified. The missing data

indicator could also moderate associations in the regression model, in which case the estimates and conclusions about robustness could change.

The Blimp output also includes tables of regression model parameters for auxiliary variables and incomplete predictors. The auxiliary variable models appear in `OUTCOME MODEL ESTIMATES` section with the focal results, and the auto-generated predictor models are displayed under the heading `PREDICTOR MODEL ESTIMATES`. Section 6.2 includes a summary table from one of these supporting models. These additionally results are not of substantive interest and would not be reported.

References

Asparouhov, T., & Muthén, B. (2010a). Bayesian analysis using Mplus: Technical implementation.

Asparouhov, T., & Muthén, B. (2010b). Chi-square statistics with multiple imputation. Retrieved 2/4/2016, from

Asparouhov, T., & Muthén, B. (2021). Advances in Bayesian model fit evaluation for structural equation models. *Structural Equation Modeling: A Multidisciplinary Journal*, *28*(1), 1–14. https://doi.org/doi.org/10.1080/10705511.2020.1764360

Barnard, J., & Rubin, D. B. (1999). Small-sample degrees of freedom with multiple imputation. *Biometrika*, *86*(4), 948–955. https://doi.org/DOI 10.1093/biomet/86.4.948

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Erlbaum.

Collins, L. M., Schafer, J. L., & Kam, C. M. (2001). A comparison of inclusive and restrictive strategies in modern missing data procedures. *Psychological Methods*, *6*(4), 330–351. https://doi.org/10.1037/1082-989X.6.4.330

Enders, C. K. (2022). *Applied Missing Data Analysis* (2nd ed.). Guilford Press.

Enders, C. K., Du, H., & Keller, B. T. (2020). A model-based imputation procedure for multilevel regression models with random coefficients, interaction effects, and other nonlinear terms. *Psychological Methods*, *25*(1), 88–112. https://doi.org/10.1037/met0000228

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2014). *Bayesian data analysis* (3rd ed.). CRC Press.

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, *7*, 457–472. https://doi.org/10.1214/ss/1177011136

Graham, J. W. (2003). Adding missing-data-relevant variables to FIML-based structural equation models. *Structural Equation Modeling: A Multidisciplinary Journal*, *10*(1), 80–100. https://doi.org/10.1207/S15328007sem1001_4

Grund, S., Robitzsch, A., & Lüdke, O. (2023). Package 'mitml'. https://cran.r-project.org/web/packages/mitml/mitml.pdf

Keller, B. T. (2024). *rblimp: Integration of the Blimp Software Into R*. In (Version 0.1.31) Retrieved from https://github.com/blimp-stats/rblimp.

Keller, B. T., & Enders, C. K. (2021). Blimp user's guide (Version 3). www.appliedmissingdata.com/blimp

Levy, R., & McNeish, D. (2023). Perspectives on Bayesian inference and their implications for data analysis. *Psychological Methods*, *28*(3), 719–739. https://doi.org/doi.org/10.1037/met0000443

Li, K. H., Raghunathan, T. E., & Rubin, D. B. (1991). Large-sample significance levels from multiply imputed data using moment-based statistics and an F reference distribution. *Journal of the American Statistical Association*, *86*(416), 1065–1073. https://doi.org/Doi 10.2307/2290525

Little, R. J. A., & Rubin, D. B. (2020). *Statistical analysis with missing data* (3rd ed.). Wiley.

Montague, M., Enders, C., & Castro, M. (2005). Academic and behavioral outcomes for students at risk for emotional and behavioral disorders. *Behavioral Disorders*, *31*(1), 84–94.

Montague, M., Krawec, J., Enders, C., & Dietz, S. (2014). The effects of cognitive strategy instruction on math problem solving of middle-school students of varying ability. *Journal of Educational Psychology*, *106*(2), 469–481.

Rights, J. D., & Sterba, S. K. (2019). Quantifying explained variance in multilevel models: An integrative framework for defining R-squared measures. *Psychological Methods*, *24*, 309–338. https://doi.org/dx.doi.org/10.1037/met0000184

van Buuren, S. (2018). *Flexible imputation of missing data* (2nd ed.). Chapman and Hall.