Blimp and `rblimp` Quick Start Guide

Blimp and `rblimp` are free software tools available at www.appliedmissingdata.com/blimp. A detailed user guide with dozens of analysis examples is available here. The native graphical interface, Blimp Studio, runs on macOS and Windows, with command-line versions available for Linux. The `rblimp` R package allows researchers to interact with Blimp through R, providing access to R's scripting and visualization capabilities, including plotting functions not available in the graphical interface. Because `rblimp` uses the Blimp computational engine, the native application must be installed before installing the R package. The online materials at https://github.com/blimp-stats/fsem include detailed installation instructions.

Model specification uses a simple scripting language that requires no prior knowledge of factored regression or Bayesian estimation. A typical Blimp script consists of several major commands (ending with a colon), each with its own subcommands or keywords (ending with a semicolon). Broadly speaking, most Blimp commands fall into three categories: those that describe variables and their attributes (e.g., `DATA`, `VARIABLES`, `MISSING`, `ORDINAL`, `NOMINAL`, `COUNT`, `LATENT`), define the analysis model (e.g., `CENTER`, `MODEL`, `SIMPLE`, `PARAMETERS`), or control the MCMC algorithm (e.g., `BURN`, `ITER`, `SEED`).

To illustrate, corresponds an analysis where a three-categorical nominal predictor moderates the influence of a latent factor. In line with standard procedures from linear regression (Aiken & West, 1991; Cohen et al., 2002), the structural model includes products of the dummy codes and the latent variable.

$$Y_i = \beta_{0Y} + \beta_{1Y}(\eta_i) + \beta_{2Y}(M_{1,i}) + \beta_{3Y}(M_{2,i}) + \beta_{4Y}(\eta_i)(M_{1,i}) + \beta_{5Y}(\eta_i)(M_{2,i}) + \varepsilon_{Y,i} = \bar{Y}_i + \varepsilon_{Y,i}$$

The Blimp script for the analysis is shown below.

```
DATA: example.dat;            # raw text file
VARIABLES: y x1 x2 x3 x4;     # variable names
MISSING: 999;                 # missing value code
ORDINAL: x1:x4;               # define ordinal variables
NOMINAL: m;                   # define multicategorical variables
LATENT: eta;                  # define latent variable
MODEL:                        # regression models
      eta ~~ eta@1;           # factor variance
      eta ~ m;                # latent on multicategorical predictor
      eta -> x1@lo1 x2:x4;    # measurement model
```

```
      x3 ~~ x4;              # correlated residual
      y ~ eta m eta*m;       # structural model with latent interaction
SIMPLE: eta | m;             # simple intercepts and slopes
BURN: 10000;                 # warm-up iterations
ITER: 10000;                 # analysis iterations
SEED: 90291;                 # random number seed
```

A brief description of several major Blimp commands follows, not all of which appear in the previous script. The software's User Guide has detailed descriptions of all Blimp commands, including many not shown here, along with dozens of analysis examples.

DATA: The DATA command specifies the input data set, which must be saved as a .csv (comma separated values) format or a whitespace (including tab) delimited file (e.g., .dat or .txt). Blimp accepts only numeric characters for data values (e.g., a nominal variable cannot have alphanumeric labels as score values), although alphanumeric characters (e.g., NA) can be used for missing value codes. Variable names can appear in the column headers, but the VARIABLES command (described next) must be omitted. No file path is needed if the Blimp script (the .imp file) is located in the same directory as the data.

VARIABLES: The VARIABLES command specifies the variable names for the data set listed on the DATA command in the input file. This command should not be used if the data file has variable names as column headers. The variable list may include variables that are not used in an analysis.

MISSING: The MISSING command is used to specify the missing value code. Missing values can be coded with a single numeric (e.g., 999) or alphanumeric value (e.g., NA).

ORDINAL: The ORDINAL command identifies binary or ordinal variables that appear in a MODEL statement. By default, Blimp adopts a probit link.

NOMINAL: The NOMINAL command specifies nominal variables that appear in a MODEL statement. Nominal variables must be represented as a single variable with numeric codes. Blimp automatically recodes the discrete responses into a set of dummy codes during estimation, where the first (lowest) code is the reference category. By default, Blimp adopts a probit or logit link function depending on whether the variable is an exogeneous predictor or dependent variable.

COUNT: The COUNT command identifies count variables that appear in a MODEL statement. By default, Blimp adopts a negative binomial link function.

LATENT: The LATENT command is used to define latent variables (e.g., factors in a measurement model) that will be referenced in the MODEL section. Latent variables can be specified at any level of a multilevel model. This specification references cluster-level identifier variables from the CLUSTERID line.

CLUSTERID: The CLUSTERID command specifies cluster-level identifier variable(s) needed for a multilevel analysis or multilevel imputation. Two-level analyses require a single

identifier for the level-2 sampling unit (cluster), and three-level analyses require level-2 and level-3 identifier variables. The order of the identifier variables does not matter, as Blimp automatically determines variable levels.

CENTER: The `CENTER` command is used to center predictor variables in regression equations. Predictor variables in a multilevel regression model can be centered at the grand means or latent group means.

MODEL: The `MODEL` command typically consists of one or more regression models. Blimp's modeling framework can accommodate a wide range of analyses ranging from basic multiple regression models to complicated multilevel structural equation models with interactions involving latent variables.

SIMPLE: The `SIMPLE` command is used to request conditional effects (e.g., simple intercepts and simple slopes) from a regression model with an interaction effect. When using `rblimp`, the `simple_plot()` function can be used to graph simple intercepts and slopes from a model with an interaction, and the `jn_plot()` function produces Johnson–Neyman regions of significance. The `PARAMETERS` command can also be used to compute contrasts.

BURN: The `BURN` command specifies the number of burn-in or warm-up iterations. Each MCMC chain (two by default) completes the specified number of iterations before saving estimates for the final parameter summaries.

ITER: The `ITER` command specifies the total number of iterations for the analysis summary tables. The total number of iterations is distributed equally across the number of MCMC chains (the default is two).

SEED: The `SEED` command specifies the random number seed for MCMC estimation. this value must be an integer.

With the exception of variable attributes stored in the R data frame and missing data code, the commands in a Blimp Studio script correspond to input arguments in the `rblimp()` function.

```
mymodel <- rblimp(                  # rblimp function call
      data = example                # R data frame
      ordinal = 'x1:x4',            # define ordinal variables
      nominal = 'm',                # define multicategorical variables
      latent = 'eta',               # define latent variable
      model = '                     # regression models
      eta ~~ eta@1;                 # factor variance
      eta ~ m;                      # latent on predictor
      eta -> x1@lo1 x2:x4;          # measurement model
      x3 ~~ x4;                     # correlated residual
      y ~ eta m eta*m;',            # model with latent interaction
      simple = 'eta | m',           # simple intercepts and slopes
      burn = 10000,                 # warm-up iterations
```

```
I     ter = 10000,                  # analysis iterations
      seed = 90291                  # random number seed
)

# print output
output(mymodel)

# plot simple slopes
simple_plot(y ~ eta | m.1 + m.2, mymodel)
```

The `rblimp` package has several graphing functions that are not available in Blimp Studio graphical interface. The `simple_plot()` function in the last line is one example, which graphs the simple slopes of the latent variable for each categorical group.