

***ParcelAlloc.sas* Manual: Detecting and Correcting for Parcel-Allocation Variability**

Introduction

This manual provides directions for implementing procedures recommended in Sterba and MacCallum (2010) for detecting, reporting, and correcting for the effects of parcel-allocation variability on model fit and parameter estimates. Conditions under which parcel-allocation variability is maximized—that is, where reporting and correcting for parcel-allocation variability are most important—are detailed in Sterba and MacCallum (2010).

This manual describes a *two-step* strategy for detecting/correcting for parcel-allocation variability. The *first step* is to run a SAS Macro (*ParcelAlloc.sas*) that takes a single user-input item-level dataset and randomly generates many item-to-parcel allocations, producing a set of parcel-level data sets that are saved to a directory. The *second step* is to run Mplus Monte Carlo code that batch-analyzes the set of parcel-level datasets in that directory, and produces aggregated output that tells the user: (a) the average estimates for each structural parameter and model fit index (i.e., estimates “controlling for” parcel-allocation variability) and (b) the standard deviation of each structural parameter and model fit index across its parcel-allocation distribution (i.e., the amount of parcel-allocation variability per estimate). Both (a) and (b) are useful quantities to report in a journal article when a parcel analysis has been performed.

In the first two sections of this manual, instructions are provided for implementing the SAS Macro and the Mplus Monte Carlo code (both available at <http://www.unc.edu/~ssterba/parcel.htm>). In the last section of this manual, three fully-worked examples are discussed; the complete set of input and output files for these examples is also provided at <http://www.unc.edu/~ssterba/parcel.htm>.

Step 1: Running the SAS Macro *ParcelAlloc.sas*

For Step 1, the user needs to provide an *item-level dataset*. The user also needs to obtain and save the SAS Macro *ParcelAlloc.sas*, and needs to adjust the input specifications for *ParcelAlloc.sas* exclusively in the file *Commandlines-parcelalloc.sas*, as discussed below. Both *ParcelAlloc.sas* and *Commandlines-parcelalloc.sas* are downloadable from <http://www.unc.edu/~ssterba/parcel.htm>.

(a) In *Commandlines-parcelalloc.sas* first specify the number of factors that will have at least one parcel indicator. Factors that are involved in the to-be-analyzed statistical model but that have no indicators in need of parceling should *not* be counted. In this example code, there are two factors that will contain at least one parcel indicator each.

```
%let factors=2;
```

(b) For each factor in (a), specify the number of items that will be used to form parcels on that factor. For example, if 9 items loaded on one factor, but only 6 of these items needed to be parceled, the user would specify 6 here. Whereas, if 9 items loaded on one factor and all 9 needed to be parceled, the user would specify 9 here. Note that different factors can have different numbers of items-to-be-parceled. Ensure that every factor included in (a) has a corresponding `numitemsfac` statement in (b). In this example code, the first factor has 12 items to be parceled, and the second factor has 10 items to be parceled.

```
%let numitemsfac1=12;  
%let numitemsfac2=10;
```

(c) For each factor in (a), specify the total number of parcels that will be constructed for that factor. Note that different factors can have different numbers of parcels. Ensure that every factor included in (a) has a corresponding `numparcelsfac` statement. In this example code, the 12 items for factor 1 are to be made into 4 parcels and the 10 items on factor 2 are to be made into 3 parcels.

```
%let numparcelsfac1= 4;  
%let numparcelsfac2= 3;
```

(d) For each factor in (a), specify a vector of parcel-sizes (i.e., #items/parcel for each parcel). Note that there can be different #items/parcel (i.e., different parcel sizes) for each parcel, if desired. In this example code, the vector of parcel-sizes for factor 1, called `varlist1` contains four parcel-sizes, one for each of the parcels listed in (c): the first parcel on factor 1 has 3 items/parcel; the second parcel on factor 1 has 2 items/parcel, the third parcel on factor 1 has 3 items/parcel; and the fourth parcel on factor 1 has 4 items/parcel. Since in (c) we specified `numparcelsfac1=4`, the corresponding vector of parcel-sizes, `varlist1`, has 4 elements; since in (c) we specified `numparcelsfac2=3`, the corresponding vector of parcel-sizes, `varlist2`, has 3 elements. Furthermore, the *sum* of each `varlist` vector should equal the `numitemsfac` from (b).

NOTE: Do not include commas between elements of the varlist vector.

```
%let varlist1= 3 2 3 4;  
%let varlist2= 3 5 2 ;
```

(e) Specify the sample size for the input item-level dataset.

```
%let n= 75;
```

(f) Specify the Directory to which the generated parcel-allocation datasets should be saved. This also initializes a new permanent SAS library called *Parcel* at this directory location. **NOTE: Do not specify the file name, only specify the directory name, and ensure it ends with a backslash.**

```
%let directory= C:\;
```

(g) Next, specify the name of the input SAS item-level dataset. This SAS dataset can either be a ‘working’ SAS dataset or a ‘permanent’ SAS dataset from an already-initialized SAS library. If the input SAS dataset was saved to the directory in (f), then you need to specify it as: `Parcel.yourdataset` because the directory in (f) refers to the library *Parcel*. **IMPORTANT REQUIREMENTS FOR INPUT DATASET:** In order to use the *ParcelAlloc* macro, the user needs to supply an item-level dataset that contains all variables involved in the to-be-analyzed statistical model. Some variables involved in the statistical model may need to be parceled; others may not.

Variables that do need to be parceled must be identified via the following variable naming convention. For factor Q , its k items to-be-parceled would be named `fQitem1 ... fQitemk`. To illustrate, if 12 items on factor 1 needed to be parceled, these items should be named:

`flitem1, flitem2, flitem3, flitem4, flitem5, flitem6, flitem7, flitem8, flitem9, flitem10, flitem11, flitem12.`

And if 10 items on factor 2 needed to be parceled, these items should be named:

`f2item1, f2item2, f2item3, f2item4, f2item5, f2item6, f2item7, f2item8, f2item9, f2item10.`

Variables that do not need to be parceled will be included as-is in every outputted parcel-dataset, and will be included as-is in the final, estimated model in *Mplus*. These not-to-be-parceled variables can retain their original names (e.g., `gender`, `race`, `age`) (see Example 2). Or, not-to-be-parceled items loading on a factor can be given names according to the variable naming-convention—*as long as* their item numbers consecutively follow those of the to-be-parceled items loading on that factor (see Example 1).

```
%let inputitemdataset= parcel.youritemdat;
```

(h) Specify the missing data identifier in the input dataset provided in (f). The missing data identifier can be nonnumeric (e.g., “.”) or numeric (e.g., -999).

```
%let missing=.;
```

(i) Specify the number of random parcel-allocations to be generated. The user must specify at least three 3 parcel-allocations. **NOTE:** For larger sample sizes and larger numbers of items to-be-parceled, generation of parcel-allocations takes longer. Messages in the log window indicate to the user when each additional allocation completes. We recommend performing a time trial with a few parcel-allocations to ascertain how many parcel-allocations are practically feasible for the dataset at hand using this macro. It is often desirable to try two different numbers of allocations to see if results are sensitive to the number of allocations requested. **NOTE:** Generated parcel-allocations are saved as .dat files in the Directory specified in (f) above. For Z generated parcel-allocations, the parcel-allocation datasets are named: `parcelruns1.dat...parcelrunsZ.dat`.

```
%let arep= 100;
```

(j) Specify where the SAS Macro *ParcelAlloc.sas* was saved.

```
%include "c:\ParcelAlloc.sas";
```

Frequently-asked questions about the *ParcelAlloc.sas* Macro:

- *What kind of items are suitable for use with ParcelAlloc.sas?*


As reviewed in Sterba and MacCallum (2010), a common practice in the parceling literature is to parcel items that are unidimensional. Additionally, the performance of this Macro has, to date, only been evaluated for normally distributed items.

- *Does the parcel-allocation macro accommodate missing data?*

Yes; if more than one, but less than all, of the items in a parcel are missing for case *i*, then the parcel score is calculated by averaging over only the non-missing items. If all of the items in a parcel are missing for case *i*, the parcel score will be missing for case *i*. Input datasets for the three worked Examples involve different patterns of missing data. However, the consequence of missing data, and various types of missing data (e.g., missing-at-random, missing-completely-at-random, missing-not-at-random; monotone missingness, nonmonotone missingness, etc.) for parcel-allocation variability has not yet been evaluated.

- *Can I see how the parceled variables and other variables in the output parcel-allocation datasets are ordered and labeled?*

When the macro is finished running, the names of the variables contained in each parcel-allocation output dataset (i.e., parcelruns1.dat – parcelrunsZ.dat) are printed to the SAS LOG, in the order (top to bottom) in which these variables appear in the output datasets. In the below log excerpt, we can see that in each parcel-allocation output dataset the variables appear in the following order: f1parc1 f1parc2 f1parc3 f1parc4 f2parc1 f2parc2 f2parc3.



The screenshot shows the SAS Log window titled "SAS - [Log - (Untitled)]". The menu bar includes File, Edit, View, Tools, Solutions, Window, and Help. Below the menu bar is a toolbar with icons for opening, saving, printing, and other functions. The log text displays the following message:

```
The variables in each parcelruns data file appear in this order
1 f1parc1
2 f1parc2
3 f1parc3
4 f1parc4
5 f2parc1
6 f2parc2
7 f2parc3
```

Importantly, note that you may see a SAS Warning message in your log file above this list, to the effect of:

WARNING: The query as specified involves ordering by an item that doesn't appear in its SELECT clause. Since you are ordering the output of a SELECT DISTINCT it may appear that some duplicates have not been eliminated.

This Warning is not of concern for the present purposes and can be ignored. (It is just telling us that SAS SQL sorted variable labels by a numerical variable corresponding to their order in the outputted datasets, and then printed the list of just those variable labels—without including the numerical sorting variable. If you are interested in

obtaining more information about this warning message, and why it is not of concern for our purposes, please see http://analytics.ncsu.edu/sesug/2006/CC01_06.PDF.)

- *Can I see which items were allocated to which parcels for a given parcel-allocation?*

The Macro saves a permanent dataset called *Allocations.sas* to the directory specified in (f). In this dataset, each row corresponds to a separate allocation; the allocation-number (i.e., repetition) is found in the last column to the right. Hence, if 100 allocations were requested in (d), *Allocations.sas* will contain 100 rows. Note that this *Allocations.sas* file is for reference purposes only. It is not needed for further analyses and can be disregarded if desired.

Here is an excerpt from *Allocations.sas* for the present example.

VIEWTABLE: Parcel.Allocations													
	f2y1	f2y2	f2y3	f2y4	f2y5	f2y6	f2y7	f2y8	f2y9	f2y10	f1y1	f1y2	
1	9	2	6	10	4	3	7	1	8	5	1	7	
2	2	10	4	8	6	9	7	3	5	1	8	10	
3	2	5	9	3	6	8	4	1	10	7	2	12	
4	10	3	6	5	9	4	1	2	8	7	6	8	
5	7	8	2	6	4	9	10	5	3	1	5	9	
6	3	9	8	2	4	1	6	5	7	10	12	11	
7	4	1	2	7	10	9	3	5	8	6	7	11	
8	10	7	4	9	8	3	2	5	6	1	6	7	
9	9	10	5	7	1	3	6	8	2	4	12	3	

Column headings indicate whether we are looking at factor 2's randomly-assigned item numbers (columns 1-10) or factor 1's randomly-assigned item numbers (columns 11-22). To interpret the contents of each row, refer to the parcel-sizes requested in (d) above. For factor 2, parcel-sizes were 3, 5, 2 (i.e. 3 items for the first parcel, 5 items for the second parcel, and 2 items for the third parcel). This means that in allocation #1 (i.e. row 1), the 3 items allocated to f2parc1 were f2item9, f2item2, f2item6, and the 5 items allocated to f2parc2 were f2item10, f2item4, f2item3, f2item7, f2item1, and the 2 items allocated to f2parc3 were f2item8 and f2item5. Because for factor 1, the parcel-sizes in (d) were 3, 2, 3, 4, the next 3 numbers in the row (not all shown) would correspond with the f1parc1, the following 2 numbers in the row (not shown) would correspond with the f1parc2, the following 3 numbers in the row (not shown) would correspond with the f1parc3, and the following 4 numbers in the row (not shown) would correspond with the f1parc4. The last number in the row is the allocation/repetition number. This same interpretation applies to row #2, row#3, etc.

- *Does the ParcelAlloc.sas macro form parcels by summing or averaging?*
ParcelAlloc.sas forms parcels only by averaging. If summing is used to form parcels in the context of missing data, cases with more missing data will result in a smaller sum, and thus a smaller parcel score.

Step 2. Running the *Mplus* Monte Carlo code.

Theoretically, any structural equation modeling program that can be executed in batch-mode could be used to compile and analyze the parcel-allocation datasets (parcelruns1.dat – parcelrunsZ.dat). Here, we demonstrate how to do this using *Mplus* 5.1 (Muthén & Muthén, 1998-2008) because *Mplus*' built-in Monte Carlo facility makes it particularly easy and user-friendly to do so.

If the user is interested only in obtaining across-allocation means and standard deviations for structural parameter estimates and model fit indices, an abbreviated *Mplus* input file called *Mpluscode_Basic.inp* is sufficient. This code is presented and discussed first. If an across-allocation mean and standard deviations of per-factor average parcel-loadings, and/or per-factor average error variances, are desired (discussed later), some additional *Mplus* code called *Mpluscode_Advanced.inp* is needed. This code is presented and discussed second.

Mpluscode_Basic.inp file for obtaining across-allocation means and SD of structural parameters and model fit:

```
TITLE:      Parcel Simulation;
DATA:      FILE IS parcelrunsreplist.dat;

TYPE IS MONTECARLO;

VARIABLE: NAMES ARE flparc1 flparc2 flparc3 flparc4
                  f2parc1 f2parc2 f2parc3;

MISSING ARE .;

USEVARIABLES flparc1 flparc2 flparc3 flparc4
              f2parc1 f2parc2 f2parc3;

ANALYSIS: TYPE = general;

MODEL:
FAC1 by flparc1* flparc2 flparc3 flparc4;
FAC2 by f2parc1* f2parc2 f2parc3;

fac1@1;
fac2@1;
fac1 with fac2;

OUTPUT: TECH9;
```

DATA statement:

- Requires **Mpluscode_Basic.inp** to be saved to the Directory specified in (f).
- Requires FILE to be specified as **parcelrunsreplist.dat**. The file parcelrunsreplist.dat was automatically saved to the Directory specified in (f) by the SAS Macro. Parcelrunsreplist.dat provides *Mplus* with a listing of all parcel datasets that need to be analyzed.
- **TYPE IS MONTECARLO** must be written on the DATA statement.

VARIABLE statement:

- NAMES ARE statement *must* provide the variable names listed in the SAS LOG by the Macro—in the order given in the SAS LOG. See FAQ in section 1 and also worked Examples for further details.
- MISSING DATA ARE statement must specify the same missing data code as was specified in *Commandlines-parcelalloc.sas*. If a non-numeric missing data code such as “.” was specified in *Commandlines-parcelalloc.sas*, then specify: MISSING DATA ARE . ; If a numeric missing data code such as -999 was specified in *Commandlines-parcelalloc.sas* then specify MISSING DATA ARE ALL (-999).

ANALYSIS statement:

- Can be modified in any way by user.

MODEL statement:

- Can be modified in any way by user.

OUTPUT statement:

- Needs to include **TECH9**. Adding ‘Standardized’ produces standardized parameter estimates.

Here are excerpts and interpretation of relevant *Mpluscode_Basic.out* output:

```
INPUT READING TERMINATED NORMALLY
SUMMARY OF ANALYSIS

Number of groups                                1
Average number of observations                  75

Number of replications
  Requested                                    100
  Completed                                    100

Number of dependent variables                   7
Number of independent variables                 0
Number of continuous latent variables           2

Input data file(s)
Multiple data files from
parcelrunsreplist.dat
```

Here we can see that 100 random parcel-allocations were analyzed and all 100 converged; therefore across-allocation estimates were compiled across all 100 allocations.

```
TECHNICAL 9 OUTPUT

Errors for replication with data file parcelruns1.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns2.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns3.dat:
```

THE MODEL ESTIMATION TERMINATED NORMALLY

WARNING: THE RESIDUAL COVARIANCE MATRIX (THETA) IS NOT POSITIVE DEFINITE.
THIS COULD INDICATE A NEGATIVE VARIANCE/RESIDUAL VARIANCE FOR AN OBSERVED
VARIABLE, A CORRELATION GREATER OR EQUAL TO ONE BETWEEN TWO OBSERVED
VARIABLES, OR A LINEAR DEPENDENCY AMONG MORE THAN TWO OBSERVED VARIABLES.
CHECK THE RESULTS SECTION FOR MORE INFORMATION.
PROBLEM INVOLVING VARIABLE F2PARC2.

Errors for replication with data file parcelruns4.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns5.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns6.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns7.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns8.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns9.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns10.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns11.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

Errors for replication with data file parcelruns12.dat:

THE MODEL ESTIMATION TERMINATED NORMALLY

CONTINUED THROUGH DATA FILE PARCELRUNS100.DAT

In the above excerpt, we see that a few of the parcel-allocations resulted in improper solutions. Sterba and MacCallum (2010) found that allocation-variability results for parameter estimates and model fit did not differ meaningfully if these improper allocations were included or excluded. If the user wishes to exclude these replications, the user would need to delete the improper allocation datasets (e.g. parcelruns3.dat) from the Directory specified in (f) and delete their corresponding rows in parcelrunsreplist.dat, and then re-run *Mpluscode_Basic.inp*.

TESTS OF MODEL FIT

Number of Free Parameters	22
Chi-Square Test of Model Fit	
Degrees of freedom	13
Mean	16.134

	Std Dev	5.417
CFI		
	Mean	0.927
	Std Dev	0.076
TLI		
	Mean	0.915
	Std Dev	0.164
AIC		
	Mean	1001.475
	Std Dev	24.276
BIC		
	Mean	1052.460
	Std Dev	24.276
RMSEA		
	Mean	0.050
	Std Dev	0.041
SRMR		
	Mean	0.067
	Std Dev	0.012

Above we have the across-allocation mean of each fit index, and the across-allocation standard deviation of each fit index. For this sample we could report that, “controlling for parcel-allocation variability by averaging across 100 parcel-allocations, we found acceptable fit according to the SRMR and RMSEA and chi-square, but not the CFI and TLI. Results for a single parcel-allocation were not reported because from allocation-to-allocation fit could range from excellent according to all fit indices, to not-close or even poor according to all fit indices.” Across-allocation SD of the BIC or AIC could be useful in determining whether BIC or AIC differences *between* alternative models are greater or less than parceling-induced BIC or AIC differences *within* each model’s parcel-allocations.

MODEL RESULTS								
		Population	ESTIMATES Average	Std. Dev.	S. E. Average	M. S. E.	95% Cover	% Sig Coeff
FAC1	WITH							
FAC2		0.000	0.4154	0.0877	0.1742	0.1802	0.240	0.760

The above excerpt from the parameter estimates output shows the one structural parameter estimate in this model (the factor covariance). We are interested in the columns labeled *ESTIMATES Average*, *Std. Dev.*, and *% Sig. Coeff.* For this sample we could report that, “controlling for parcel-allocation variability by averaging across 100 parcel-allocations, the inter-factor covariance was .42. Results for a single parcel allocation were not reported because there was considerable across-allocation variability in the estimate (68% of allocations fell within $\pm .09$ of .42). Further, in 76% of allocations the factor covariance was significant, and in 24% of allocations it was nonsignificant. The across-allocation average of the analytic standard error was .17.”

[The columns labeled *Population*, *MSE*, *95% Cover*. are not of use to us; they are automatically produced by the software for use with traditional simulation studies where population parameters are known and repeated samples are generated from a population.]

If the word “Standardized” were added to the Output line of the code, all of the above would also have been reported for the factor correlation. See Sterba & MacCallum (2010) for some discussion of differences between standardized and unstandardized parcel-solutions.

In this parameter estimates excerpt, we did not show across-allocation means and SD for each parcel’s factor loading or each parcel’s error variance. These quantities are not very meaningful to report because, for prespecified numbers of items/factor, parcels/factor, items/parcel, they *have to* vary allocation-by-allocation when loadings and/or error variances are unequal, due to the reshuffling of indicator reliabilities—even if sampling error is zero (Yuan, Bentler, & Kano, 1997). The *average* of the loadings per factor and the *average* of error variances per factor, on the other hand, vary allocation-by-allocation only to the extent that sampling error drives across-allocation variability (see Sterba & MacCallum, 2010). Thus, it is often of interest to see how much the latter quantities vary across allocations. In order to obtain the mean and SD of the per-factor average loading and per-factor average error variance, the following modifications to the input file would be required:

Mpluscode_Advanced.inp file for obtaining across-allocation means and SD of structural parameters, model fit, and per-factor average loadings and average error variances.

```
TITLE:      Parcel Simulation;
DATA:      FILE IS parcelrunsreplist.dat;

TYPE IS MONTECARLO;

VARIABLE: NAMES ARE flparc1 flparc2 flparc3 flparc4
f2parc1 f2parc2 f2parc3;

MISSING ARE .;

USEVARIABLES flparc1 flparc2 flparc3 flparc4
f2parc1 f2parc2 f2parc3;

ANALYSIS: TYPE = general;
MODEL:

FAC1 by
flparc1* (F1L1)      !bold= naming loadings on fac1
flparc2 (F1L2)
flparc3 (F1L3)
flparc4 (F1L4);

FAC2 by
f2parc1* (F2L1)      !bold= naming loadings on fac2
f2parc2 (F2L2)
f2parc3 (F2L3);

flparc1 (F1EV1);      !bold= naming err variances
flparc2 (F1EV2);
flparc3 (F1EV3);
```

```

f1parc4 (F1EV4);
f2parc1 (F2EV1);
f2parc2 (F2EV2);
f2parc3 (F2EV3);

fac1@1;
fac2@1;
fac1 with fac2;

MODEL CONSTRAINT:
NEW (F1LAVG F2LAVG F1EVAVG F2EVAVG);      !names new parameters
F1Lavq=(F1L1 + F1L2 + F1L3 + F1L4)/4;      !computes new parms
F2Lavq=(F2L1 + F2L2 + F2L3)/3;
F1Eavg=(F1EV1 + F1EV2 + F1EV3 + F1EV4)/4;
F2Eavg=(F2EV1 + F2EV2 + F2EV3)/3;

OUTPUT: TECH9;

```

That is, in order to get *Mplus* to calculate the allocation-variation in the per-factor average loading and per-factor average error variance, the user needs to first name each of the parcel loadings and each of the parcel error variances by putting an arbitrary name in parentheses following that parameter estimate (here, F1L1-F1L4; F2L1-F2L3; F1EV1-F1EV4; F2EV1-F2EV3). Next, the user needs to manually create per-factor average loadings and per-factor average error variances in the MODEL CONSTRAINT section of the *Mplus* code. The per-factor average estimates (here named F1LAVG, F2LAVG, F1EVAVG, F2EVAVG) are named using the NAME statement within MODEL CONSTRAINT. **NOTE:** In this example, we identified the model by fixing the variance of each factor to 1. If one wanted to identify the model by fixing an anchor parcel-indicator for each factor to 1, then one would need to put 1 in place of that parcel-indicator loading in the calculations for the Model Constraint statement.

This additional code gives us the following added piece of relevant output:

New/Additional Parameters							
	Population	ESTIMATES		S. E.	M. S. E.	95%	% Sig
		Average	Std. Dev.	Average		Cover	Coeff
F1LAVG	0.500	0.3097	0.0184	0.0461	0.0366	0.000	1.000
F2LAVG	0.500	0.3995	0.0236	0.0533	0.0107	0.590	1.000
F1EVAVG	0.500	0.3000	0.0237	0.0315	0.0406	0.000	1.000
F2EVAVG	0.500	0.2825	0.0388	0.0439	0.0488	0.010	0.970

These are unstandardized estimates. Here we can say that “controlling for parcel-allocation variability by averaging across 100 parcel-allocations, the per-factor average loading was .31 for factor 1 and .40 for factor 2, and the per-factor average error variance was .30 for factor 1 and .28 for factor 2. Results for a single parcel allocation were not reported because of across-allocation variability in these estimates (standard deviations of per-factor average loadings across-allocations were approximately .02).”

Final Notes about *Mplus* Code:

If parcel variances are very small (<.001), *Mplus* TYPE=MONTECARLO may give an error message and it may be necessary to use other software to compile results. This is a characteristic of *Mplus*, not a function of how the SAS data are read out (scientific notation or not, Fortran formatting, number of decimal points, etc.) nor a function of how *Mplus* reads in the data (free format or fixed format with Fortran descriptor). The Macro reads out data for the parcelruns1.dat-parcelrunsZ.dat files with 19 digits total including the decimal place and six digits after the decimal.

Example Files.

On our website, along with the *ParcelAlloc.sas* macro, the following files are provided for each of three fully-worked examples:

Step 1.

- Item-level input dataset
- Input specifications (*Commandlines-parcelalloc.sas*)
- Output parcel-allocation datasets (parcelruns1.dat – parcelrunsZ.dat)
- Screenshot of SAS Log showing variable-labels in output parcel-allocation data
- Concatenated table of allocations (*Allocations.sas*) **(for reference purposes only)**
- Output list of parcel-allocation datasets for *Mplus* (parcelrunreplist.dat)

Step 2.

- *Mplus* Monte Carlo input code (*Mpluscode_Advanced.inp*)
- *Mplus* Monte Carlo output code (*Mpluscode_Advanced.out*)

These three examples were presented to illustrate how the Macro handles common data situations that arise in practice. Unique aspects of results interpretation are highlighted for each. Each example includes the extended *Mplus* code to create per-factor average loadings and per-factor average error variances; if these are not desired, the *Mplus* code can be simplified as described earlier in the documentation.

Example 1.

Features of Example 1 are:

- Two-factor CFA model where all of the indicators of factor 1 are parcels and all but 2 of the indicators of factor 2 are parcels. That is, factor2 has two item indicators (labeled f2item11, f2item12) that are not-to-be-parceled.
- Numeric missing data code (-999)
- unequal #items/factor
- unequal #parcel indicators/factor
- unequal #items/parcel

Important points to note:

- Even though factor 2 starts with 12 item indicators, because 2 of the 12 are left unparceled, *Commandlines-parcelalloc_Ex1.sas* only specifies that there are 10 to-be-parceled items for factor 2.
- The SAS LOG excerpt shows that the first two variables in each of the parcelruns1.dat – parcelruns100.dat files are the designated not-to-be parceled

items from factor 2 (f2item11, f2item12). Note in *Mpluscode_AdvancedEx1.inp* that these two variables were therefore placed first on the VARIABLE: NAMES ARE line.

- Inspection of the parcelruns1.dat – parcelruns100.dat files show that, as intended, these first two variables (the f2item11 and f2item12 not-to-be-parceled items) are included as-is from the input file (*Example1.sas*), and that these first two variables do not change from the parcelruns1.dat file to the parcelruns100.dat file. Whereas the remaining variables (which are parcels) do, naturally, change from parcelruns1.dat to parcelruns100.dat (i.e., allocation to allocation).
- The *Mpluscode_AdvancedEx1.out* file shows that, although the two unparceled item indicators (f2item11, f2item12) *do not change* from allocation to allocation, their factor loadings *do change* from allocation to allocation (as illustrated by the non-zero across-allocation standard deviations for these estimates). This is because the correlations of these two item-indicators with other parcel-indicators of factor 2 do change across allocations.

Example 2.

Features of Example 2 are:

- Two-factor SEM model where all of the indicators for Factor1 and Factor2 are parcels, however, there are 3 not-to-be-parceled covariates that predict Factor2. These covariates are arbitrarily named x1, x2, x3.
- Non-numeric missing data code (.)
- Unequal #items/factor
- Unequal #parcels indicators/factor
- Equal #items/parcel

Important points to note:

- The SAS LOG excerpt shows that the first three variables in each of the parcelruns1.dat – parcelruns100.dat files are the designated not-to-be-parceled covariates predicting factor 2 (x1, x2, x3). Note in *Mpluscode_AdvancedEx2.inp* that these three variables were therefore placed first on the VARIABLE: NAMES ARE line.
- Inspection of the parcelruns1.dat – parcelruns100.dat files show that, as intended, these first three variables (the x1, x2, x3 not-to-be-parceled covariates) are included as-is from the input file (*Example2.sas*), and that these first three variables do not change from parcelruns1.dat file to parcelruns100.dat file, whereas the rest of the variables (which are parcels) do, naturally, change from parcelruns1.dat to parcelruns100.dat (i.e., allocation-to-allocation).
- The *Mpluscode_AdvancedEx2.out* file shows that, although the covariances among the 3 unparceled predictors of factor 2 (x1, x2, x3) *do not change* from

allocation to allocation (all have across-allocation SD = 0), the regressions of factor 2 on x1, x2, and x3 *do change* from allocation to allocation (as illustrated by the non-zero across-allocation standard deviations for these regression estimates). Scrolling down to the R-SQUARE portion of the output, we can see that, relatedly, the proportion of explained variance in the endogenous factor 2 changes from allocation-to-allocation (nonzero SD). This is partially a function of the fact that the nature of factor 2 changes across allocations, because parcels and their intercorrelations change across allocations.

Example 3.

Features of Example 3 are:

- Three-factor CFA model where all of the indicators of Factor1 and Factor2 are parcels, but *none* of the indicators of Factor3 are parcels. The third factor has 5 item indicators (named f3item1-f3item5) that are not-to-be-parceled.
- Non-numeric missing data code (.)
- Unequal #items/factor
- Unequal #parcels/factor
- Unequal #items/parcel
- Because Factor3 contains no parcel indicators, *Commandlines-parcelalloc_Ex3.sas* specifies that there are only 2 factors containing parcel indicators.
- The SAS LOG excerpt shows that the first five variables in each of the parcelruns1.dat – parcelruns100.dat files are the designated not-to-be parceled items from factor 3 (f3item1-f3item5). Note in *Mpluscode_AdvancedEx3.inp* that these five variables were therefore placed first on the VARIABLE: NAMES ARE line.
- Inspection of the parcelruns1.dat – parcelruns100.dat files show that, as intended, these first five variables (f3item1-f3item5) are included as-is from the input file (*Example3.sas*), and that these first five variables do not change from the parcelruns1.dat file to parcelruns100.dat file. Whereas the rest of the variables (which are parcels) do, naturally, change from parcelruns1.dat to parcelruns100.dat (i.e., allocation-to-allocation).
- *Mpluscode_AdvancedEx3.out* file shows that, following intuition, there is more allocation-to-allocation variability in the per-factor average parcel loading for factor 1 and factor 2 than there is allocation-to-allocation variability in the (unparceled) item loadings for factor 3. Nonetheless, the item loadings on factor 3 are not immune to allocation-variability; there is still a small amount of allocation-variability in these loadings even though factor 3 does not involve any parcels. Also following intuition, there is more allocation-to-allocation variability

in the R-squared of the parcel-indicators than in the R-squared of the item-indicators. Relatedly, there is more allocation-to-allocation variability in the correlation among the two factors with parcel indicators (factor 1 and factor 2) than there is in the correlation between an item-indicator factor (factor 3) and either of the parcel-indicator factors (factor 1 or factor 2).

- Note that the TECH9 output indicates that a few parcel-allocation repetitions resulted in improper solutions. These repetitions were left in for this example, but users may consider removing them (see earlier discussion).