Adversarial Simulation

A case study

Leonard Kosanke

The git hash is: be58c

```
library(knitr)
library(kableExtra)
library(tidyverse)
```

Deviations from the simulation protocol

Across studies

For pragmatic purposes of the simulation implementation, I was unable to track and include non-convergence explicitly and exclude cases where it occurred.

Studies 1,2,3

For all these studies, 3 deviations were present. Firstly, the total number of conditions turned out to be higher, as the calculation in the protocol forgot the addition of N=50 to be added to the factor sample size. Secondly, It was not detailed what type of LSAM estimation will be implemented. For completeness sake, I included ML- as well as ULS-LSAM estimation.

Study 1b

In Study 1b, contrary to the simulation protocol, I added another sample size with N=50 to get a more nuanced understanding of the small-sample bias of LSAM. For the same reasons

as in studies 1,2 and 3, I used ML- as well as ULS-LSAM estimation. Calculation of the relative bias was not feasible in this study, as for the conditions where phi was 0, dividing by 0 would have lead to infinite values.

Study 4

Again, I forgot the addition of N=50 in the original condition calculation. Importantly, the model parameters and resulting population-level covariance matrices for the different DGM's provided by Robitzsch (2022) in their github repository were not the same as in the first paper by Rosseel and Loh (2022): They omitted, for example, the cross-loading of the last factor in DGM2, without an apparent reason. Additionally, they had different values for the residual variances without giving insight into why they chose to do so and how (there was no calculation in the simulation script provided). For these reasons, and because of the fact that the calculation of the values in Rosseel and Loh (2022) was replicable, I chose to use the values of Rosseel and Loh (2022) by generating them with the same functions they used. In consequence, a fourth cross-loading was present in DGM 2. The only difference in my data-simulation is that I only used the 3 out of 4 misspecification conditions that were present in Rosseel and Loh (2022). Importantly, due to the implementation of the functions in the paper by Rosseel and Loh (2022), the naming of the DGM's changed. In their paper, what Robitzsch (2022) called DGM1 was implemented with a value of 0. DGM 2 and 3 were thus also implemented with values 1 and 2. To avoid confusion, I used the terminology of the code from Rosseel and Loh (2022). Thus, the naming was different when interpreting the results.

Study 4a

I decided to include a variation of study 4a after all, as the relation between beta/phi values and N was interesting: The original paper showed that results differed between estimators, depending on the size of the factor correlation in Studies 5, 6 and 1b, as well as differential effects due to N in Studies 1,2 and 3. I deemed it interesting to investigate these results in a model with a different number of factors. These findings, combined with our goal to investigate realistic data conditions, rather than just the population level, left me to include Study 4a, with the addition of an additional factor sample size. As DGM1 neither contained misspecification (which is central to our research question), nor did it lead to interesting

results in the original study 4, it was omitted. Similarly, as no substantial conclusions were drawn with regards to RMSE in the original papers study 1 and 2, I omitted its calculation as well. Lastly, for the same reasons as argued for Study 4, I again created the DGM's based on the functions used in Rosseel and Loh (2022). As in Study 4, from now on I will also adhere to the naming that follows from the code taken from Rosseel and Loh (2022), so that DGM's 1-3 are now DGM's 0-2.

Studies 5,6

As we were not interested in the comparative performance of SAM vs. SEM at the population level for our research question, but in the performance comparison under realistic conditions, especially in small to moderate sample sizes, I decided to not replicate Studies 5 and 6 after all. I deemed this a fair deviation from my simulation protocol, especially because no substantial performance differences arose in the results of these studies, and the most relevant aspect for us, the differential effect due to the size of factor correlation, is already covered in two different models in studies 1b and 4a.

Result analysis

Original paper

Bias

In the original paper, Robitzsch (2022) had some inconsistencies when it came to the interpretation of the results of their simulation studies. They did not thoroughly define an a prior criterium for when an estimator outperformed another. Only in their Studies 5 and 6, they defined an absolute bias equal to or larger than 0.05 as relevant. Additionally, when an estimator had a minimum percentage error of 20% higher than another, they defined it as being better than the other, in these studies. Across the other studies, interpretations sometimes varied. In some studies (e.g. Study 1), they seemed to orient on the same value of 0.05, but this time as an absolute difference value, without the addition of a certain percentage cut-off. In other studies (e.g. Study 2), they seemed to deviate from this value to define relevant difference in absolute bias. Importantly, this deviation was present in both

directions, and therefore did not appear to be consciously biased towards one direction of possible outcomes. In consequence, it appears that interpretations aren't coherent across the studies.

Another important aspect to highlight in the original papers results refers to the general presence of a negative small sample bias in LSAM estimation. This bias makes LSAM appear to perform superior in conditions with positive values in unmodelled cross-loadings and residual correlation, but worse when the values are negative. Even though this bias was correctly identified and explained already in Study 1, Robitzsch (2022) conduct multiple follow-up studies only investigating positive values in unmodelled cross-loadings and residual correlation, namely in study 3. This is done without a general note of some sorts, that this bias should apply in these data-generating scenarios the same as in Studies 1 and 2. In consequence, the interpretations Robitzsch (2022) make, are somewhat misleading on first sight, if one does not account for this bias. This does not, however, explain why they did not include more studies or conditions with negatively valenced residual correlations or cross-loadings. I hypothesize, that the goal of the paper was to show that even in these favorable constellations, SAM does not perform better than SEM.

To be fair, it should be stated that most of the final conclusions drawn in the paper are correct, anyway. By saying that SAM- should not "generally" be preferred over SEM-estimation, they are right in that there are many data constellations, where SAM does not outperform traditional SEM. They also acknowledge, that based on their results for non-saturated structural models, SAM seems to be the better choice. Unfortunately, they use the term "generally" in a second context, when they argue in the discussion that still, SAM should still generally be used. Here, however, they say it should be used to avoid confounding the definition of measurement models from the estimation of structural models. This, according to Robitzsch (2022), often happens in applied research by including cross-loadings and residual correlations at will, without substantive reasons. In this sense, SAM should generally be used.

For my studies, in order to have a more coherent interpretation of results, I defined an estimator to outperform another, based on the cut-off of 0.05 for bias. Additionally, I only used the term "generally" to refer to the first meaning, that it outperforms across multiple data generating scenarios or conditions.

SD and RMSE

Even though Robitzsch (2022) reported tables with results for SD and RMSE, they rarely explicitly mentioneded them in the results sections or gave cut-offs for substantial relevance.

SD was only reported and mentioned in the conditions without misspecification in Study 1, even though also claimed to be calculated in studies 2 and 3. In this study, they mentioned a 6.6% reduction of SD for LSAM in comparison to ML. This calculation, however, was wrong as the reduction amounts to 8%. For my interpretation, I reported percentage reductions larger than 5% as well and compared them with the original study, where possible.

RMSE (and average RMSE) was reported in Studies 1-4, but only in studies 2 and 4 explicitly interpreted for my estimators of interest. In study 2, differences of up to 0.03 were present and interpreted as not being substantial differences. In study 4, differences of 0.01 were present and interpreted as slight efficiency gains. I infered that this lack of mentioning and consistency can be interpreted as there being no differences of interest. Thus, I expected my studies to yield similar results. If, however, there were differences larger than 0.03, I reported them.

Recalculation of Confidence intervals

I did multiple mistakes with the calculation of the confidence intervals, as I did not include the absolute, but relative mean in the two limits for studies 1b, 4 and 4a. Also, the standard error was wrongly calculated as relative. For Studies 1, 2 and 3, I misscalculated the standard error, as well as the confidence interval by including absolute values in their calculation, even though I was interested in relative values in these studies. Thus, I had to reextract the results for bias correctly, which was done in postprocess_results.R. All I did there, was reapply the corrected extract_results() function and the same report_bias() function from the individual simulations, one after another. For the results of bias, I loaded these processed files, instead of the original ones, from the newly created folder: SimulationResultsProcessed. For presentation purposes, we did not include the CI's in the tables for this document. They can, however, be easily included as they are contained in the metrics_list object in each study in the same folder.

Results Simulation 1: 2-factor-CFA with 0,1 or 2 correlated residuals

Error, warnings and messages

First, lets check for any messages, warnings or errors:

```
#Save all errors, warnings and messages
all_messages_s1 <- readRDS("SimulationResults/sim1_results_error.rds")


#errors
errors_s1 <- all_messages_s1$errors
errors_sum_s1 <- unlist(errors_s1)
length(errors_sum_s1)
```

```
[1] 0
```

```
#warnings
warnings_s1 <- all_messages_s1$warnings
warnings_sum_s1 <- unlist(warnings_s1)
length(warnings_sum_s1)
```

```
[1] 245
```

```
#messages
messages_s1 <- all_messages_s1$messages
messages_sum_s1 <- unlist(messages_s1)
length(messages_sum_s1)
```

```
[1] 0
```

No errors and messages were present. There were, however, 245 warnings, I will investigate in detail.

Warnings investigation

Investigating the 'warnings' object list, it became apparent that all the warnings were the same: `no non-missing arguments to min; returning Inf`.

```
unique(warnings_s1)
```

```
[[1]]
character(0)

[[2]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"

[[3]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"

[[4]]
[1] "no non-missing arguments to min; returning Inf"

[[5]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"
[5] "no non-missing arguments to min; returning Inf"
[6] "no non-missing arguments to min; returning Inf"
```

This warning could indicate either convergence issues or improper solutions due to small sample size in one or multiple estimators. Unfortunately, due to the implementation of the study, I was unable to check for more details, as whether there is any kind of pattern with

regards to the warning occurrence. As the 245 are less than 0.5% of the 52500 estimations performed, I will continue with the interpretation as planned, as their impact is negligible.

Bias

```
# Load the relative bias results
bias_ci_s1 <- readRDS("SimulationResultsProcessed/sim1_rel_bias_ci.rds")

shorten_names <- function(df) {
  df$method_metric <- sub("_rel_bias", "", df$method_metric)
  return(df)
}

# Function to create a styled table for each condition
create_styled_table <- function(data, condition) {
  data <- shorten_names(data)
  data <- data %>%
    mutate(across(where(is.numeric), ~ round(., 3)))

  # Format table
  kbl(data,
      col.names = c("Method", "50", "100", "250", "500",
                    "1000", "2500", "1e+05"), booktabs = TRUE) %>%
    kable_styling(full_width = F, position = "left",
                  latex_options =c("scale_down", "hold_position")) %>%
    add_header_above(c(" " = 1, "Sample Size" = 7)) %>%
    row_spec(0, bold = TRUE) %>%
    kable_classic(full_width = F) %>%
    add_header_above(c("Condition:" = 8), bold = TRUE,  align = "l",
                     line = TRUE, italic = TRUE) %>%
    footnote(general = paste("Condition:", condition))
}
```

Condition without correlated residuals (0_0.12)

```
create_styled_table(bias_ci_s1[["0_0.12"]], "0_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	-0.045	-0.011	0.001	-0.003	0.003	-0.002	-0.000
SEM_ULS	0.024	0.022	0.012	0.002	0.006	-0.001	-0.000
LSAM_ML	-0.394	-0.270	-0.111	-0.056	-0.022	-0.011	-0.000
LSAM_ULS	-0.393	-0.270	-0.111	-0.056	-0.022	-0.011	-0.000
GSAM_ML	-0.394	-0.270	-0.111	-0.056	-0.022	-0.011	-0.000
GSAM_ULS	-0.393	-0.270	-0.111	-0.056	-0.022	-0.011	-0.000

Note:

Condition: 0_0.12

Both standard SEM estimators were unbiased (i.e. relative bias < 0.05) across all conditions of N. The 4 SAM estimators were negatively biased for sample sizes up to 500 and had nearly identical estimation results. Across all estimators, the relative bias decreased with increasing N.

Thus, SEM estimation outperformed SAM in small to moderate sample sizes, when there was no misspecification.

Condition with one negative residual correlation (1_-0.12)

```
create_styled_table(bias_ci_s1[["1_-0.12"]], "1_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	-0.120	-0.091	-0.074	-0.083	-0.074	-0.079	-0.077
SEM_ULS	-0.056	-0.059	-0.067	-0.080	-0.074	-0.081	-0.080
LSAM_ML	-0.451	-0.327	-0.187	-0.144	-0.110	-0.100	-0.090
LSAM_ULS	-0.450	-0.327	-0.187	-0.144	-0.110	-0.100	-0.090
GSAM_ML	-0.451	-0.327	-0.187	-0.144	-0.110	-0.100	-0.090
GSAM_ULS	-0.449	-0.327	-0.187	-0.144	-0.110	-0.100	-0.090

Note:

Condition: 1_-0.12

All estimators were negatively biased (i.e. relative bias >= -0.05) across all N conditions. Across all estimators except SEM-ULS, the bias decreased substantially with increasing N. For SEM-ULS, it increased slightly but unsubstantially up to N=500, but was lowest overall. The 4 SAM estimators had nearly identical estimation results. In relative comparison, the two SEM estimators outperformed all SAM estimators in every condition of N, even though this difference was not substantive for N=1000 and higher.

Thus, SEM estimation outperformed SAM in small to moderate sample sizes, when there was one unmodelled negative residual correlation.

Condition with one positive residual correlation (1_0.12)

```
create_styled_table(bias_ci_s1[["1_0.12"]], "1_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.041	0.090	0.109	0.104	0.111	0.106	0.109
SEM_ULS	0.103	0.118	0.111	0.100	0.105	0.098	0.100
LSAM_ML	-0.343	-0.211	-0.027	0.032	0.066	0.077	0.089
LSAM_ULS	-0.342	-0.211	-0.027	0.032	0.066	0.077	0.089
GSAM_ML	-0.343	-0.211	-0.027	0.032	0.066	0.077	0.089
GSAM_ULS	-0.342	-0.211	-0.027	0.032	0.066	0.077	0.089

Note:

Condition: 1_0.12

Besides SEM-ML in N=50, both SEM estimators were positively biased across all N, with no visible trend for increasing N. All SAM estimators had nearly identical values and increasing relative bias for increasing N. They were negatively biased up to N=250 and positively biased for the other N conditions. For N=250 and N=500, their relative bias was unsubstantial. In relative comparison, the two SEM estimators outperformed all SAM estimators for sample sizes up to 100. For all higher N, the SAM estimators outperformed the SEM estimators, even though the difference was unsubstantial starting from N=1000.

Thus, SEM estimation outperformed SAM in smaller samples (N=50-100), whereas SAM outperformed SEM in moderate samples (N250-500), and slightly in higher sample sizes, when there was one unmodelled positive residual correlation.

Condition with two negative residual correlations (2_-0.12)

```
create_styled_table(bias_ci_s1[["2_-0.12"]], "2_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	-0.205	-0.175	-0.166	-0.168	-0.161	-0.166	-0.164
SEM_ULS	-0.139	-0.145	-0.159	-0.167	-0.163	-0.170	-0.169
LSAM_ML	-0.498	-0.385	-0.272	-0.225	-0.196	-0.189	-0.180
LSAM_ULS	-0.497	-0.385	-0.272	-0.225	-0.196	-0.189	-0.180
GSAM_ML	-0.497	-0.385	-0.272	-0.225	-0.196	-0.189	-0.180
GSAM_ULS	-0.496	-0.385	-0.272	-0.225	-0.196	-0.189	-0.180

Note:

Condition: 2_-0.12

All estimators were negatively biased across all N conditions. Across all estimators except SEM-ULS, the bias decreased substantially with increasing N. For SEM-ULS, it slightly, but unsubstantially increasd up to N=500, but was lowest overall in these conditions. The 4 SAM estimators had nearly identical estimation results. In relative comparison, the two SEM estimators outperformed all SAM estimators in every condition of N, even though this difference was not substantive for N=1000 and higher.

Thus, SEM estimation outperformed SAM in small to moderate sample sizes, when there were two unmodelled negative residual correlations.

Condition with two positive residual correlation (2_0.12)

```
create_styled_table(bias_ci_s1[["2_0.12"]], "2_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.105	0.172	0.199	0.191	0.198	0.193	0.195
SEM_ULS	0.182	0.194	0.201	0.189	0.194	0.188	0.189
LSAM_ML	-0.299	-0.135	0.055	0.113	0.154	0.166	0.179
LSAM_ULS	-0.298	-0.134	0.055	0.113	0.154	0.166	0.179
GSAM_ML	-0.299	-0.135	0.055	0.113	0.154	0.166	0.179
GSAM_ULS	-0.297	-0.134	0.055	0.113	0.154	0.166	0.179

Note:

Condition: 2_0.12

Both SEM estimators were positively biased across all N, with no visible trend for increasing N for ULS, but a substantial increase for ML. All SAM estimators had nearly identical values and increasing relative bias for increasing N. They were negatively biased up to N=100 and positively biased for the other N conditions. In relative comparison, the two SEM estimators outperformed all SAM estimators for N=50. For all higher N, the SAM estimators outperformed the SEM estimators (substantially, besides ML in N=100), even though the difference was unsubstantial starting from N=1000.

Thus, SEM estimation outperformed SAM in small samples (N=50), whereas SAM outperformed SEM in small to moderate samples (N100-500), and slightly in higher sample sizes when two unmodelled positive residual correlations were present.

Summary

in 0.5% of estimations, a warning referring to potential problems with convergence or improper solutions due to small sample sizes in one or multiple estimators occurred. In conditions with no, one or two negative residual correlations, SEM outperformed SAM in small to moderate sample sizes. This seemed to be independent of the amount of misspecification (1 vs. 2). Comparing the conditions with one and two positive residual correlations, the results suggested that SAM outperformed SEM in smaller to moderate samples, with increasing

amount of misspecifiation (even better for 2 than for 1 unmodelled residual correlation). In small samples, SEM tended to perform better.

Comparison with original paper

Comparing these results with the original paper, the results were similar and can be seen as mostly replicated. The only difference lied in the potential convergence issues in a small number of estimations. Besides that, I drew the same conclusions as Robitzsch (2022): All SAM approaches appeared to be generally negatively biased for small to moderate samples in the 2-factor-CFA model. This leads to the comparatively worse performance in the conditions with no, one or two negative residual correlations. Additionally, this negative bias corrects for unmodelled positive residual correlations and makes SAM appear superior in these conditions.

SD

```
# Load the relative bias results
sd_s1 <- readRDS("SimulationResults/sim1_sd.rds")
```

No residual correlations (0_0.12)

```
create_styled_table(sd_s1[["0_0.12"]], "0_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.299	0.208	0.122	0.083	0.060	0.037	0.006
SEM_ULS_sd	0.276	0.194	0.118	0.082	0.059	0.037	0.006
LSAM_ML_sd	0.232	0.190	0.123	0.084	0.059	0.037	0.006
LSAM_ULS_sd	0.233	0.190	0.123	0.084	0.059	0.037	0.006
GSAM_ML_sd	0.235	0.190	0.123	0.084	0.059	0.037	0.006
GSAM_ULS_sd	0.237	0.190	0.123	0.084	0.059	0.037	0.006

Note:

Condition: 0_0.12

Across all estimators, the standard deviation decreased with increasing N. There were only two substantial differences: There was a substantial percentage reduction of SD up to 22.4% when comparing SEM-ML and LSAM-ML in N=50, as well as a substantial percentage reduction of SD up to 8.6% when comparing SEM-ML and LSAM-ML in N=100.

Thus, SAM outperformed SEM in terms of SD reduction in smaller samples, when there was no misspecification.

One negative residual correlation (1_-0.12)

```
create_styled_table(sd_s1[["1_-0.12"]], "1_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.321	0.213	0.125	0.085	0.061	0.038	0.006
SEM_ULS_sd	0.303	0.198	0.120	0.083	0.060	0.038	0.006
LSAM_ML_sd	0.233	0.187	0.123	0.086	0.061	0.039	0.006
LSAM_ULS_sd	0.234	0.187	0.123	0.086	0.061	0.039	0.006
GSAM_ML_sd	0.235	0.187	0.123	0.086	0.061	0.039	0.006
GSAM_ULS_sd	0.236	0.187	0.123	0.086	0.061	0.039	0.006

Note:

Condition: 1_-0.12

Across all estimators, the standard deviation decreased with increasing N. There were only two substantial differences: A substantial percentage reduction of SD up to 27.4% when comparing SEM-ML and LSAM-ML in N=50, as well as a substantial percentage reduction of SD up to 12.2% when comparing SEM-ML and LSAM-ML in N=100.

Thus, SAM outperformed SEM in terms of SD reduction in smaller samples when one unmodelled negative residual correlation was present.

One positive residual correlation (1_0.12)

```
create_styled_table(sd_s1[["1_0.12"]], "1_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.289	0.197	0.120	0.083	0.058	0.037	0.006
SEM_ULS_sd	0.268	0.181	0.116	0.081	0.058	0.036	0.006
LSAM_ML_sd	0.240	0.196	0.126	0.083	0.060	0.037	0.006
LSAM_ULS_sd	0.240	0.196	0.126	0.083	0.060	0.037	0.006
GSAM_ML_sd	0.243	0.196	0.126	0.083	0.060	0.037	0.006
GSAM_ULS_sd	0.242	0.196	0.126	0.083	0.060	0.037	0.006

Note:

Condition: 1_0.12

Across all estimators, the standard deviation decreased with increasing N. The SAM estimators outperformed both SEM estimators in N=50 with a percentage reduction of SD up to 16.9%. ULS-SEM outperformed all SAM estimators with a percentage reduction of SD up to 8.16% in N=100-250. No substantial differences between the two SEM and all SAM estimators arose, starting from moderate sample sizes (N=500-2500).

Thus, SAM outperformed SEM in terms of SD reduction in small samples, whereas SEM outperformed SAM in moderate samples, when one positive unmodelled residual correlation was present.

Two negative residual correlations (2_-0.12)

```
create_styled_table(sd_s1[["2_-0.12"]], "2_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.327	0.219	0.125	0.087	0.062	0.038	0.006
SEM_ULS_sd	0.316	0.204	0.120	0.085	0.061	0.038	0.006
LSAM_ML_sd	0.232	0.191	0.123	0.089	0.062	0.039	0.006
LSAM_ULS_sd	0.233	0.191	0.123	0.089	0.062	0.039	0.006
GSAM_ML_sd	0.234	0.191	0.123	0.089	0.062	0.039	0.006
GSAM_ULS_sd	0.235	0.191	0.123	0.089	0.062	0.039	0.006

Note:

Condition: 2_-0.12

Across all estimators, the standard deviation decreased with increasing N. The SAM estimators outperformed both SEM estimators in N=50 with a percentage reduction of SD up to 29% and in N=100 with up to 12.8%. For higher N, there were no substantial differences.

Thus, SAM outperformed SEM in terms of SD reduction in smaller samples.

Two positive residual correlations (2_0.12)

```
create_styled_table(sd_s1[["2_0.12"]], "2_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.282	0.187	0.118	0.081	0.057	0.036	0.005
SEM_ULS_sd	0.254	0.176	0.114	0.079	0.057	0.035	0.005
LSAM_ML_sd	0.239	0.198	0.126	0.082	0.059	0.037	0.006
LSAM_ULS_sd	0.239	0.198	0.126	0.082	0.059	0.037	0.006
GSAM_ML_sd	0.240	0.198	0.126	0.082	0.059	0.037	0.006
GSAM_ULS_sd	0.241	0.198	0.126	0.082	0.059	0.037	0.006

Note:

Condition: 2_0.12

Across all estimators, the standard deviation decreased with increasing N. The SAM estimators outperformed both SEM estimators in N=50 with a percentage reduction of SD up to 15.2%. Both SEM estimators ourperformed the SAM estimators in N=100 with a percentage reduction of SD up to 11.1% and in N=250 with up to 10.3%. For higher N, there was no substantial differences.

Thus, SAM outperformed SEM in terms of SD reduction in small samples, but SEM outperformed SAM in smaller to moderate samples.

Summary and comparison with original paper

Overall, SAM outperformed SEM in terms of SD reduction in small to smaller samples, especially for negatively correlated residuals. In smaller to moderate samples, SEM outperformed SAM only for positively correlated residuals. For conditions with no misspecification, which were the only conditions reported in Robitzsch (2022), the percentage reduction favoring SAM was nearly identical for N=100, and the reduction seems to be even more significantly in favor of SEM in the smaller sample size.

RMSE

```
rmse_s1 <- readRDS("SimulationResults/sim1_rmse.rds")
```

No residual correlations (0_0.12)

```
create_styled_table(rmse_s1[["0_0.12"]], "0_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.301	0.208	0.122	0.083	0.060	0.037	0.006
SEM_ULS_rmse	0.277	0.194	0.118	0.082	0.059	0.037	0.006
LSAM_ML_rmse	0.331	0.250	0.140	0.090	0.061	0.038	0.006
LSAM_ULS_rmse	0.331	0.250	0.140	0.090	0.061	0.038	0.006
GSAM_ML_rmse	0.333	0.250	0.140	0.090	0.061	0.038	0.006
GSAM_ULS_rmse	0.334	0.250	0.140	0.090	0.061	0.038	0.006

Note:

Condition: 0_0.12

Across all estimators, the RMSE decreased with increasing N. In N=100, both SEM estimators were more efficient than all SAM estimators. Besides that, there were no substantial differences present.

One negative residual correlation (1_-0.12)

```
create_styled_table(rmse_s1[["1_-0.12"]], "1_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.329	0.220	0.132	0.098	0.075	0.061	0.047
SEM_ULS_rmse	0.305	0.201	0.127	0.096	0.075	0.062	0.048
LSAM_ML_rmse	0.357	0.271	0.166	0.122	0.090	0.072	0.054
LSAM_ULS_rmse	0.357	0.271	0.166	0.122	0.090	0.072	0.054
GSAM_ML_rmse	0.358	0.271	0.166	0.122	0.090	0.072	0.054
GSAM_ULS_rmse	0.358	0.271	0.166	0.122	0.090	0.072	0.054

Note:

Condition: 1_-0.12

Across all estimators, the RMSE decreased with increasing N. For small to moderate samples (N=50-250), both SEM estimators were more efficient than all SAM estimators. For higher N, there were no substantial differences.

One positive residual correlation (1_0.12)

```
create_styled_table(rmse_s1[["1_0.12"]], "1_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.290	0.204	0.137	0.104	0.089	0.074	0.066
SEM_ULS_rmse	0.275	0.194	0.134	0.101	0.085	0.069	0.060
LSAM_ML_rmse	0.316	0.233	0.127	0.085	0.072	0.059	0.054
LSAM_ULS_rmse	0.316	0.233	0.127	0.085	0.072	0.059	0.054
GSAM_ML_rmse	0.318	0.233	0.127	0.085	0.072	0.059	0.054
GSAM_ULS_rmse	0.317	0.233	0.127	0.085	0.072	0.059	0.054

Note:

Condition: 1_0.12

Across all estimators, the RMSE decreased with increasing N. For smaller samples (N=50-100), SEM-ULS was more efficient than all SAM estimators. For higher N, there were no substantial differences present.

Two negative residual correlations (2_-0.12)

```
create_styled_table(rmse_s1[["2_-0.12"]], "2_-0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.350	0.243	0.159	0.134	0.115	0.107	0.099
SEM_ULS_rmse	0.326	0.222	0.153	0.131	0.115	0.109	0.102
LSAM_ML_rmse	0.378	0.300	0.204	0.162	0.133	0.120	0.108
LSAM_ULS_rmse	0.378	0.300	0.204	0.162	0.133	0.120	0.108
GSAM_ML_rmse	0.379	0.300	0.204	0.162	0.133	0.120	0.108
GSAM_ULS_rmse	0.379	0.300	0.204	0.162	0.133	0.120	0.108

Note:

Condition: 2_-0.12

Across all estimators, the RMSE decreased with increasing N. For small to moderate samples (N=50-500), The two SEM estimators were more efficient than all SAM estimators. For higher N, there were no substantial differences present.

Two positive residual correlations (2_0.12)

```
create_styled_table(rmse_s1[["2_0.12"]], "2_0.12")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.289	0.214	0.168	0.140	0.132	0.121	0.117
SEM_ULS_rmse	0.276	0.211	0.166	0.139	0.129	0.118	0.114
LSAM_ML_rmse	0.299	0.214	0.131	0.107	0.109	0.106	0.108
LSAM_ULS_rmse	0.298	0.214	0.131	0.107	0.109	0.106	0.108
GSAM_ML_rmse	0.300	0.214	0.131	0.107	0.109	0.106	0.108
GSAM_ULS_rmse	0.299	0.214	0.131	0.107	0.109	0.106	0.108

Note:

Condition: 2_0.12

Across all estimators, the RMSE decreased with increasing N. For moderate samples (N=250-500), all SAM estimators were more efficient than the two SEM estimators. For higher and lower N, there were no substantial differences present.

Summary

Overall, classic SEM estimation was more efficient than SAM estimation in small to moderate samples. Only in the case of two positive residual correlation, SAM was more efficient than SEM in moderate samples.

Summary Study 1

I gained the following insights with regards to the 2-factor-CFA model:

In terms of bias in small to moderate samples, SEM outperformed SAM in conditions with no, one and two negative residual correlations. SAM appeared to outperform SEM for positive residual correlations. This, however, was an artifact of its negative small sample bias and not its performance. The results align with the ones obtained by the original paper.

With regards to SD and RMSE, results were incoherent with no clear pattern emerging.

Thus, it cannot be stated that one estimation method was generally performing better than the other, based on the results of Study 1.

Results Simulation 1b: 2-factor-CFA with 2 residual correlations and varying lambda and phi

Error, warnings and messages

First, I checked for any messages, warnings or errors the same way as in Study 1:

```
#Save all errors, warnings and messages
all_messages_s1b <- readRDS("SimulationResults/sim1b_results_error.rds")


#errors
errors_s1b <- all_messages_s1b$errors
errors_sum_s1b <- unlist(errors_s1b)
length(errors_sum_s1b)
```

```
[1] 0
```

```
#warnings
warnings_s1b <- all_messages_s1b$warnings
warnings_sum_s1b <- unlist(warnings_s1b)
length(warnings_sum_s1b)
```

```
[1] 414
```

```
#messages
messages_s1b <- all_messages_s1b$messages
messages_sum_s1b <- unlist(messages_s1b)
length(messages_sum_s1b)
```

```
[1] 0
```

No errors and messages were present. There were, however, 414 warnings, I investigated in detail.

Warnings investigation

Investigating the 'warnings' object list, it became apparent that all the warnings were the same: `no non-missing arguments to min; returning Inf`.

```
unique(warnings_s1b)
```

```
[[1]]
character(0)

[[2]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"

[[3]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
```

As before, this warning could indicate either convergence issues or improper solutions due to small sample size in one or multiple estimators. Similarly, I was unable to check for more details, besides the fact that the 414 estimation were around 0.5% of the 75000 estimations performed. I continued with the interpretation as planned, as their impact was negligible.

Bias

```
#Load processed results
bias_ci_s1b <- readRDS("SimulationResultsProcessed/sim1b_abs_bias_ci.rds")
```

Conditions with N=50

```
#Normal print
#print(bias_ci_s1b[["N_50"]][["LSAM_ML"]])
#print(bias_ci_s1b[["N_50"]][["LSAM_ULS"]])

#Formatted tables
kbl(bias_ci_s1b[["N_50"]][["LSAM_ML"]],
    col.names = c("Lambda", "phi=0", "phi=0.2",
                  "phi=0.4", "phi=0.6", "phi=0.8"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Phi values" = 5)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 1b: Absolute Bias of LSAM-ML for N=50" = 6),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 1b: Absolute Bias of LSAM-ML for N=50

	Phi values				
Lambda	**phi=0**	**phi=0.2**	**phi=0.4**	**phi=0.6**	**phi=0.8**
0.4	0.202	0.202	0.258	0.346	0.444
0.5	0.187	0.179	0.203	0.245	0.305
0.6	0.176	0.165	0.166	0.170	0.190
0.7	0.164	0.150	0.139	0.122	0.116
0.8	0.150	0.135	0.119	0.098	0.074

```
kbl(bias_ci_s1b[["N_50"]][["LSAM_ULS"]],
    col.names = c("Lambda", "phi=0", "phi=0.2",
                  "phi=0.4", "phi=0.6", "phi=0.8"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
```

```
  add_header_above(c(" " = 1, "Phi values" = 5)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 1b: Absolute Bias of LSAM-ULS for N=50" = 6),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 1b: Absolute Bias of LSAM-ULS for N=50

	Phi values				
Lambda	**phi=0**	**phi=0.2**	**phi=0.4**	**phi=0.6**	**phi=0.8**
0.4	0.204	0.203	0.258	0.345	0.440
0.5	0.188	0.180	0.203	0.244	0.304
0.6	0.176	0.165	0.166	0.170	0.190
0.7	0.164	0.150	0.139	0.122	0.116
0.8	0.150	0.135	0.119	0.098	0.074

Comparing LSAM-ML and LSAM-ULS, the values were nearly identical. Therefore, I interpretated the estimators together: Across all conditions, both estimators were biased. For higher values of lambda, the estimators had less absolute bias. This difference increased for higher phi values. For lower values of lambda, the absolute bias increased with increasing phi. This trend reversed for higher values of lambda, where the absolute bias decreased with higher phi values.

Conditions with N=100

```
kbl(bias_ci_s1b[["N_100"]][["LSAM_ML"]],
    col.names = c("Lambda", "phi=0", "phi=0.2",
                  "phi=0.4", "phi=0.6", "phi=0.8"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Phi values" = 5)) %>%
  row_spec(0, bold = TRUE) %>%
```

```
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 1b: Absolute bias of LSAM-ML for N=100" = 6),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 1b: Absolute bias of LSAM-ML for N=100

	Phi values				
Lambda	**phi=0**	**phi=0.2**	**phi=0.4**	**phi=0.6**	**phi=0.8**
0.4	0.168	0.164	0.209	0.277	0.349
0.5	0.157	0.155	0.153	0.171	0.200
0.6	0.140	0.135	0.124	0.116	0.110
0.7	0.119	0.112	0.104	0.088	0.071
0.8	0.106	0.097	0.088	0.073	0.055

```
kbl(bias_ci_s1b[["N_100"]][["LSAM_ULS"]],
    col.names = c("Lambda", "phi=0", "phi=0.2",
                  "phi=0.4", "phi=0.6", "phi=0.8"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Phi values" = 5)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 1b: Absolute bias of LSAM-ULS for N=100" = 6),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 1b: Absolute bias of LSAM-ULS for N=100					
	Phi values				
Lambda	**phi=0**	**phi=0.2**	**phi=0.4**	**phi=0.6**	**phi=0.8**
0.4	0.168	0.164	0.209	0.276	0.348
0.5	0.157	0.155	0.153	0.171	0.200
0.6	0.140	0.135	0.124	0.116	0.110
0.7	0.119	0.112	0.104	0.088	0.071
0.8	0.106	0.097	0.088	0.073	0.055

Again, the values and therefore the interpretation of LSAM-ML and LSAM-ULS were identical. Across all conditions, both estimators were biased. For higher values of lambda, the estimators had less absolute bias. This difference increased for higher phi values. For lower values of lambda, the absolute bias increased with increasing phi. This trend reversed for higher values of lambda, where the absolute bias decreased with higher phi values. Additionally, there seemed to be no differential effect due to N, as the interpretations of the different N-conditions were identical. Naturally, the bias values were larger for N=50.

Comparison to original paper

In contrast to the original paper by Robitzsch (2022), I computed absolute bias to avoid infinite values in some conditions. Additionally, computing absolute instead of relative bias solved the problem of positive and negative biases canceling each other out. Results partly aligned in terms of absolute bias: For higher values of lambda, LSAM had less absolute bias. This difference increased for higher phi values. Only for low values of lambda, the bias increased with higher values of phi. Unlike in this study, the original paper did not find a reversal effect for higher lambda values. Another difference was that we, unlike Robitzsch (2022), found estimates to be biased for low phi-values as well. This might be due to the difference of using absolute bias, thereby avoiding cancellation of positive and negative values. Still, my general conclusion corresponds to the original paper: LSAM has a general negative small sample bias. The bias is larger in magnitude for lower loadings and higher true factor correlations. Importantly, the bias seems to not disappear for low values of phi and lambda, unlike in the original paper.

Results Simulation 2: 2-factor-CFA with 1 or 2 cross-loadings

Error, warnings and messages

```
#Save all errors, warnings and messages
all_messages_s2 <- readRDS("SimulationResults/sim2_results_error.rds")


#errors
errors_s2 <- all_messages_s2$errors
errors_sum_s2 <- unlist(errors_s2)
length(errors_sum_s2)
```

```
[1] 0
```

```
#warnings
warnings_s2 <- all_messages_s2$warnings
warnings_sum_s2 <- unlist(warnings_s2)
length(warnings_sum_s2)
```

```
[1] 612
```

```
#messages
messages_s2 <- all_messages_s2$messages
messages_sum_s2 <- unlist(messages_s2)
length(messages_sum_s2)
```

```
[1] 0
```

No errors and messages were present. There were, however, 612 warnings I investigated in detail.

Warnings investigation

Again, all the warnings were the same: `no non-missing arguments to min; returning Inf`. One difference was that there were sometimes 4-6 warnings per repetition, more than before were I only saw up to 2 per repetition.

```
unique(warnings_s2)
```

```
[[1]]
character(0)

[[2]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"

[[3]]
[1] "no non-missing arguments to min; returning Inf"

[[4]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"

[[5]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"
[5] "no non-missing arguments to min; returning Inf"
[6] "no non-missing arguments to min; returning Inf"

[[6]]
[1] "no non-missing arguments to min; returning Inf"
```

```
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"

[[7]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"
[5] "no non-missing arguments to min; returning Inf"
```

As before, this warning could indicate either convergence issues or improper solutions due to small sample size in one or multiple estimators. Again, I was unable to check for more details, but the 612 estimation were only around 1.5% of the 42000 estimations performed. I continued with the interpretation as planned, as their impact was negligible.

Bias

```
# Load the relative bias results
bias_ci_s2 <- readRDS("SimulationResultsProcessed/sim2_rel_bias_ci.rds")
```

One negative cross-loading (1_-0.3)

```
create_styled_table(bias_ci_s2[["1_-0.3"]], "1_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	-0.209	-0.157	-0.134	-0.137	-0.130	-0.133	-0.130
SEM_ULS	-0.144	-0.123	-0.128	-0.140	-0.136	-0.142	-0.140
LSAM_ML	-0.535	-0.406	-0.264	-0.206	-0.173	-0.162	-0.150
LSAM_ULS	-0.534	-0.406	-0.264	-0.206	-0.173	-0.162	-0.150
GSAM_ML	-0.535	-0.406	-0.264	-0.206	-0.173	-0.162	-0.150
GSAM_ULS	-0.533	-0.406	-0.264	-0.206	-0.173	-0.162	-0.150

Note:

Condition: 1_-0.3

All estimators were negatively biased (i.e. relative bias >= -0.05) across all N conditions. Across all estimators except SEM-ULS, the bias decreased substantially with increasing N. For SEM-ULS, the bias stayed constant across N, and was lowest by a substantial marging compared to all SAM estimators up until N=500. For higher N, SEM-ML was best, but the differences between all estimators were negligible. The 4 SAM estimators have nearly identical estimation results. In relative comparison, the two SEM estimators outperformed all SAM estimators in every condition of N, even though this difference was not substantive for N=1000 and higher.

Thus, SEM estimation outperformed SAM in small to moderate sample sizes, when there was one negative cross-loading.

One positive cross-loadings (1_0.3)

```
create_styled_table(bias_ci_s2[["1_0.3"]], "1_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.160	0.190	0.205	0.205	0.212	0.207	0.208
SEM_ULS	0.202	0.204	0.200	0.195	0.199	0.193	0.193
LSAM_ML	-0.253	-0.099	0.071	0.137	0.173	0.184	0.196
LSAM_ULS	-0.251	-0.099	0.071	0.137	0.173	0.184	0.196
GSAM_ML	-0.252	-0.099	0.071	0.137	0.173	0.184	0.196
GSAM_ULS	-0.250	-0.099	0.071	0.137	0.173	0.184	0.196

Note:

Condition: 1_0.3

All estimators were biased across all N conditions. For the two SEM-estimators, the relative bias was always positive and showed no trend across N. All SAM estimators had nearly identical values and increasing relative bias for increasing N. Their bias turned from being negative for N=50-100 to positive from N=250 on. In relative comparison, the two SEM estimators outperformed all SAM estimators for N=50. For N=100-2500, the SAM estimators outperformed the SEM estimators, even though the difference was unsubstantial starting from N=1000.

Thus, SEM estimation outperformed SAM in small samples (N=50), whereas SAM outperformed SEM in smaller to moderate samples (N100-500), and slightly in higher sample sizes.

Two negative cross-loadings (2_-0.3)

```
create_styled_table(bias_ci_s2[["2_-0.3"]], "2_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	-0.420	-0.376	-0.305	-0.290	-0.270	-0.274	-0.268
SEM_ULS	-0.355	-0.320	-0.282	-0.283	-0.271	-0.279	-0.276
LSAM_ML	-0.697	-0.614	-0.482	-0.412	-0.366	-0.351	-0.335
LSAM_ULS	-0.695	-0.613	-0.482	-0.412	-0.366	-0.351	-0.335
GSAM_ML	-0.695	-0.614	-0.482	-0.412	-0.366	-0.351	-0.335
GSAM_ULS	-0.693	-0.613	-0.482	-0.412	-0.366	-0.351	-0.335

Note:

Condition: 2_-0.3

All estimators were negatively biased across all N conditions. Across all estimators, the bias decreased substantially with increasing N. The 4 SAM estimators had nearly identical estimation results. In relative comparison, the two SEM estimators substantially outperformed all SAM estimators in every condition of N.

Thus, SEM estimation outperformed SAM, when there were two negative residuals.

Two positive cross-loadings (2_0.3)

```
create_styled_table(bias_ci_s2[["2_0.3"]], "2_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.315	0.366	0.393	0.391	0.398	0.393	0.394
SEM_ULS	0.355	0.368	0.382	0.376	0.383	0.376	0.377
LSAM_ML	-0.061	0.115	0.295	0.350	0.388	0.400	0.412
LSAM_ULS	-0.060	0.115	0.295	0.350	0.388	0.400	0.412
GSAM_ML	-0.060	0.115	0.295	0.350	0.388	0.400	0.412
GSAM_ULS	-0.059	0.115	0.295	0.350	0.388	0.400	0.412

Note:

Condition: 2_0.3

All estimators were biased across N conditions. Both SEM estimators were positively biased across all N, with no visible trend for increasing N for ULS, but a substantial increase for ML. All SAM estimators had nearly identical values and increasing relative bias for increasing N. They were negatively biased in N=50 and positively biased for the other N conditions. In relative comparison, the four SAM estimators outperformed the two SEM estimators for N=50-500, even though unsubstantial for N=500. For all higher N, no estimator substantially differed from another, but the two SEM estimators slightly outperformed.

Thus, SAM outperformed SEM in small to moderate samples (N50-250).

Summary

For small to moderate samples, SEM outperformed SAM in conditions with negative cross-loadings. SAM outperformed SEM in conditions with positive cross-loadings.

Comparison with original paper

Comparing these results with the original paper, the results were similar and can be seen as replicated.

Consequently, I drew the same conclusions as Robitzsch (2022): All SAM approaches appeared to be generally negatively biased for small to moderate samples in the 2-factor-CFA

model. This lead to the comparatively worse performance in the conditions with no, one or two negative residual correlations. Additionally, this negative bias corrected for unmodelled positive cross-loadings and made SAM appear superior in these conditions.

SD

```
# Load the relative bias results
sd_s2 <- readRDS("SimulationResults/sim2_sd.rds")
```

One negative cross-loading (1_-0.3)

```
create_styled_table(sd_s2[["1_-0.3"]], "1_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.344	0.232	0.136	0.092	0.066	0.041	0.006
SEM_ULS_sd	0.336	0.220	0.131	0.090	0.064	0.040	0.006
LSAM_ML_sd	0.238	0.194	0.131	0.090	0.065	0.041	0.006
LSAM_ULS_sd	0.238	0.194	0.131	0.090	0.065	0.041	0.006
GSAM_ML_sd	0.241	0.194	0.131	0.090	0.065	0.041	0.006
GSAM_ULS_sd	0.242	0.194	0.131	0.090	0.065	0.041	0.006

Note:

Condition: 1_-0.3

Across all estimators, the standard deviation decreases with increasing N. There were only two substantial differences: One substantial percentage reduction of SD up to 30.8% when comparing SEM-ML and LSAM-ML in N=50, and one substantial percentage reduction of SD up to 16% when comparing SEM-ML and LSAM-ML in N=100.

Thus, SAM outperformed SEM in terms of SD reduction in smaller samples.

One positive cross-loading (1_0.3)

```
create_styled_table(sd_s2[["1_0.3"]], "1_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.250	0.178	0.109	0.074	0.054	0.034	0.005
SEM_ULS_sd	0.229	0.166	0.107	0.074	0.054	0.034	0.005
LSAM_ML_sd	0.234	0.192	0.122	0.081	0.057	0.035	0.005
LSAM_ULS_sd	0.235	0.191	0.122	0.081	0.057	0.035	0.005
GSAM_ML_sd	0.236	0.192	0.122	0.081	0.057	0.035	0.005
GSAM_ULS_sd	0.236	0.191	0.122	0.081	0.057	0.035	0.005

Note:

Condition: 1_0.3

Across all estimators, the standard deviation decreased with increasing N. The SAM estimators outperformed both SEM estimators in N=50 with a percentage reduction of SD up to 6.4%. ULS-SEM outperformed all SAM estimators with a percentage reduction of SD up to 14.1% in N=100, and of 12.3% in N=250 and of 8.6% in N=500. Above that point, overall SD was very small and negligible in all estimators No substantial differences between the two SEM and all SAM estimators arose starting from moderate sample sizes (N=500-2500).

Thus, SAM outperformed SEM in terms of SD reduction in small samples, whereas SEM outperformed SAM in smaller to moderate samples.

Two negative cross-loadings (2_-0.3)

```
create_styled_table(sd_s2[["2_-0.3"]], "2_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.409	0.308	0.180	0.118	0.083	0.051	0.008
SEM_ULS_sd	0.427	0.312	0.172	0.115	0.083	0.051	0.008
LSAM_ML_sd	0.249	0.205	0.144	0.106	0.077	0.049	0.008
LSAM_ULS_sd	0.251	0.205	0.144	0.106	0.077	0.049	0.008
GSAM_ML_sd	0.256	0.205	0.144	0.106	0.077	0.049	0.008
GSAM_ULS_sd	0.259	0.206	0.144	0.106	0.077	0.049	0.008

Note:

Condition: 2_-0.3

Across all estimators, the standard deviation decreased with increasing N.

The SAM estimators substantially, but decreasingly, outperformed both SEM estimators in samples N=50-500 with a SD percentage reduction of 10.1% remaining in N=500. For higher N, there were no substantial differences and overall SD was very small across estimators.

Thus, SAM outperformed SEM in terms of SD reduction in small to moderate samples.

Two positive cross-loadings (2_0.3)

```
create_styled_table(sd_s2[["2_0.3"]], "2_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.208	0.135	0.084	0.061	0.044	0.028	0.004
SEM_ULS_sd	0.175	0.131	0.085	0.062	0.045	0.029	0.004
LSAM_ML_sd	0.229	0.187	0.104	0.068	0.048	0.030	0.005
LSAM_ULS_sd	0.229	0.187	0.104	0.068	0.048	0.030	0.005
GSAM_ML_sd	0.229	0.187	0.104	0.068	0.048	0.030	0.005
GSAM_ULS_sd	0.229	0.187	0.104	0.068	0.048	0.030	0.005

Note:

Condition: 2_0.3

Across all estimators, the standard deviation decreased with increasing N.

Both SEM estimators outperform the SAM estimators in N=50-500 with a SD percentage reduction up to 11.7% remaining in N=500. For higher N, there were no substantial differences.

Thus, SEM outperformed SAM in small to moderate samples.

Summary

For negative cross-loadings, SAM tended to outperform SEM in terms of SD reduction small to moderate samples. For positive cross-loadings, SEM tended to outperform SAM in terms of SD reduction smaller to moderate samples. There seemed to be an effect due to the amount of misspecification, such that the conditions with 2 cross-loadings had larger differences in performance (in both directions).

RMSE

```
rmse_s2 <- readRDS("SimulationResults/sim2_rmse.rds")
```

One negative cross-loading (1_-0.3)

```
create_styled_table(rmse_s2[["1_-0.3"]], "1_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.366	0.250	0.158	0.123	0.102	0.090	0.078
SEM_ULS_rmse	0.347	0.232	0.151	0.123	0.104	0.094	0.084
LSAM_ML_rmse	0.400	0.312	0.206	0.153	0.123	0.105	0.090
LSAM_ULS_rmse	0.399	0.311	0.206	0.153	0.123	0.105	0.090
GSAM_ML_rmse	0.402	0.312	0.206	0.153	0.123	0.105	0.090
GSAM_ULS_rmse	0.401	0.311	0.206	0.153	0.123	0.105	0.090

Note:

Condition: 1_-0.3

Across all estimators, the RMSE decreased with increasing N. For small to moderate samples (N=50-500), both SEM estimators were more efficient than all SAM estimators. For higher N, there were no substantial differences present.

One positive cross-loading (1_0.3)

```
create_styled_table(rmse_s2[["1_0.3"]], "1_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.268	0.211	0.164	0.144	0.138	0.129	0.125
SEM_ULS_rmse	0.259	0.206	0.161	0.138	0.131	0.120	0.116
LSAM_ML_rmse	0.279	0.201	0.129	0.115	0.118	0.116	0.118
LSAM_ULS_rmse	0.279	0.200	0.129	0.115	0.118	0.116	0.118
GSAM_ML_rmse	0.280	0.201	0.129	0.115	0.118	0.116	0.118
GSAM_ULS_rmse	0.280	0.200	0.129	0.115	0.118	0.116	0.118

Note:

Condition: 1_0.3

Across all estimators, the RMSE decreased with increasing N. For smaller and large samples (N=50-100 and N=1000-100000), none of the estimators differed from each other For N=250, all 4 SAM estimators substantially outperformed the two SEM estimators. For N=500, all 4 SAM estimators substantially outperformed SEM-ML estimation.

Two negative cross-loadings (2_-0.3)

```
create_styled_table(rmse_s2[["2_-0.3"]], "2_-0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.480	0.382	0.257	0.211	0.182	0.172	0.161
SEM_ULS_rmse	0.477	0.367	0.241	0.205	0.182	0.175	0.166
LSAM_ML_rmse	0.486	0.421	0.323	0.269	0.232	0.216	0.201
LSAM_ULS_rmse	0.487	0.421	0.323	0.269	0.232	0.216	0.201
GSAM_ML_rmse	0.490	0.422	0.323	0.269	0.232	0.216	0.201
GSAM_ULS_rmse	0.490	0.421	0.323	0.269	0.232	0.216	0.201

Note:

Condition: 2_-0.3

Across all estimators, the RMSE decreased with increasing N. For small samples with N=50, none of the estimators differed in terms of efficiency. For all other conditions, The two SEM estimators were more efficient than all SAM estimators.

Two positive cross-loadings (2_0.3)

```
create_styled_table(rmse_s2[["2_0.3"]], "2_0.3")
```

Condition:							
	Sample Size						
Method	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.281	0.258	0.250	0.242	0.243	0.237	0.237
SEM_ULS_rmse	0.276	0.256	0.245	0.234	0.234	0.227	0.226
LSAM_ML_rmse	0.231	0.199	0.205	0.221	0.237	0.242	0.247
LSAM_ULS_rmse	0.232	0.199	0.205	0.221	0.237	0.242	0.247
GSAM_ML_rmse	0.232	0.199	0.205	0.221	0.237	0.242	0.247
GSAM_ULS_rmse	0.232	0.199	0.205	0.221	0.237	0.242	0.247

Note:

Condition: 2_0.3

Only for the two SEM estimators, the RMSE decreased with increasing N, but remained at higher values between 0.22-0.25 for all estimators.

For small to moderate samples (N=50-250), all SAM estimators were more efficient than the two SEM estimators. For higher N, there were no substantial differences present.

Summary

In moderate samples, SAM generally outperformed SEM in terms of RMSE. The only exception is the condition with 1 negative cross-loading, where SEM outperformed SAM in small to moderate samples. In the condition with 2 positive cross-loadings, SAM was more efficient in small samples as well. Unlike in Study 1, there seemed to be no effect due to amount of misspecification.

Summary Study 2

I gained the following insights with regards to the 2-factor-CFA model:

In terms of bias in small to moderate samples, SEM outperformed SAM in conditions with negative cross-loadings. SAM appeared to outperform SEM in conditions with positive cross-

loadings. This, however, was again an artifact of its negative small sample bias and not its performance. The results aligned with the ones obtained by the original paper.

With regards to SD reduction, findings were different. In conditions with negative cross-loadings, SAM tended to outperform SEM, whereas the opposite was true for conditions with positive cross-loadings. Here, an effect due to the amount of misspecification seemed to be present, like for RMSE in Study 1.

In terms of RMSE, results differed again. Generally, SAM was more efficient than SEM in moderate samples. For small samples, the results were not as clear. Unlike in Study 1, there seems to be no effect due to amount of misspecification.

Overall, findings are somewhat incoherent and it can not be stated that one estimation method generally performed better than the other, based on the results of Study 2.

Results simulation 3:2-factor-CFA with one cross-loading and one residual correlation

Error, warnings and messages

```
#Save all errors, warnings and messages
all_messages_s3 <- readRDS("SimulationResults/sim3_results_error.rds")


#errors
errors_s3 <- all_messages_s3$errors
errors_sum_s3 <- unlist(errors_s3)
length(errors_sum_s3)
```

```
[1] 0
```

```
#warnings
warnings_s3 <- all_messages_s3$warnings
warnings_sum_s3 <- unlist(warnings_s3)
length(warnings_sum_s3)
```

```
[1] 193
```

```
#messages
messages_s3 <- all_messages_s3$messages
messages_sum_s3 <- unlist(messages_s3)
length(messages_sum_s3)
```

```
[1] 0
```

No errors and messages were present. There were, however, 193 warnings I investigated in detail.

Warnings investigation

Investigating the 'warnings' object list, it became apparent that all the warnings were the same again: `no non-missing arguments to min; returning Inf`.

```
unique(warnings_s3)
```

```
[[1]]
character(0)

[[2]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"
[3] "no non-missing arguments to min; returning Inf"
[4] "no non-missing arguments to min; returning Inf"

[[3]]
[1] "no non-missing arguments to min; returning Inf"
[2] "no non-missing arguments to min; returning Inf"

[[4]]
[1] "no non-missing arguments to min; returning Inf"
```

As before, this warning could indicate either convergence issues or improper solutions due to small sample size in one or multiple estimators. Again, I was unable to check for more details, but as the 193 estimation amount to only around 1.8% of the 10500 estimations performed, I continued with the interpretation as planned, as their impact was negligable.

Bias

```
# Load the relative bias results
bias_ci_s3 <- readRDS("SimulationResultsProcessed/sim3_rel_bias_ci.rds")

# Round all numeric values to 3 digits
bias_ci_s3 <- bias_ci_s3 %>%
  mutate(across(where(is.numeric), ~ round(., 3)))

#Format table for results
kbl(bias_ci_s3,
    col.names = c("Method/Metric", "50", "100", "250",
                  "500", "1000", "2500", "1e+05"), booktabs = TRUE) %>%
    kable_styling(full_width = F, position = "left",
                  latex_options ="scale_down") %>%
    add_header_above(c(" " = 1, "Sample Size" = 7)) %>%
    row_spec(0, bold = TRUE) %>%
    kable_classic(full_width = F) %>%
    add_header_above(c("Study 3:" = 8), bold = TRUE,
                     align = "l", line = TRUE, italic = TRUE)
```

Study 3:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rel_bias	0.209	0.270	0.283	0.289	0.289	0.282	0.284
SEM_ULS_rel_bias	0.250	0.284	0.277	0.280	0.279	0.271	0.272
LSAM_ML_rel_bias	-0.232	-0.061	0.127	0.211	0.246	0.261	0.276
LSAM_ULS_rel_bias	-0.229	-0.060	0.127	0.211	0.246	0.261	0.276
GSAM_ML_rel_bias	-0.230	-0.060	0.127	0.211	0.246	0.261	0.276
GSAM_ULS_rel_bias	-0.228	-0.060	0.127	0.211	0.246	0.261	0.276

All estimators were biased for every condition of N. The SAM estimators had nearly identical values. Besides SEM-ULS, all estimators' bias increased substantially, even though the increase was stronger for the SAM estimators. Only the SAM estimators had negative bias for N=50-100. For N=50, none of the estimators outperformed another. For N=100-500, all SAM estimators outperformed the two SEM estimators. For higher N, none of the estimators substantially differed from another.

Thus, all SAM estimators outperformed the two SEM estimators in smaller to moderate samples.

Comparison with original paper

Comparing these results with the original paper, my results were somewhat different. Even though the results were similar for most conditions (Robitzsch (2022) had the same results in conditions N=100-250 and N=1000-100000), the results were different for N=500. This, combined with the findings from the additional condition of N=50 revealed that SAM estimation outperformed SEM estimation in this data generating scenario in small to moderate samples.

SD

```r
metrics_s3 <- readRDS("SimulationResults/sim3_metrics_list.rds")

# Load the SD results
sd_s3 <- metrics_s3[["sd"]]

# Round all numeric values to 3 digits
sd_s3 <- sd_s3 %>%
  mutate(across(where(is.numeric), ~ round(., 3)))

#Format table
kbl(sd_s3,
    col.names = c("Method/Metric", "50", "100", "250",
                  "500", "1000", "2500", "1e+05"), booktabs = TRUE) %>%
    kable_styling(full_width = F, position = "left",
                  latex_options ="scale_down") %>%
    add_header_above(c(" " = 1, "Sample Size" = 7)) %>%
    row_spec(0, bold = TRUE) %>%
    kable_classic(full_width = F) %>%
    add_header_above(c("Study 3:" = 8), bold = TRUE,
                     align = "l", line = TRUE, italic = TRUE)
```

Study 3:

	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_sd	0.243	0.173	0.110	0.077	0.055	0.034	0.005
SEM_ULS_sd	0.223	0.159	0.109	0.077	0.055	0.034	0.006
LSAM_ML_sd	0.234	0.203	0.128	0.084	0.058	0.035	0.006
LSAM_ULS_sd	0.235	0.203	0.128	0.084	0.058	0.035	0.006
GSAM_ML_sd	0.236	0.204	0.128	0.084	0.058	0.035	0.006
GSAM_ULS_sd	0.237	0.204	0.128	0.084	0.058	0.035	0.006

Across all estimators, the standard deviation decreased with increasing N. There were three

substantial differences: A substantial percentage reduction of SD up to 5.5% in favor of SEM-ULS when comparing with GSAM-ULS in N=50. Another substantial percentage reduction of SD up to 21.7% when comparing SEM-ULS when comparing with GSAM-ULS in N=100. And lastly, a substantial percentage reduction of SD up to 15.6% when comparing SEM-ULS with GSAM-ULS in N=250. For higher N, the SD still partly differed but became negligibly small.

Thus, SEM outperformed SAM in terms of SD reduction in small to moderate samples.

RMSE

```
# Load the rmse results
rmse_s3 <-  metrics_s3[["rmse"]]

# Round all numeric values to 3 digits
rmse_s3 <- rmse_s3 %>%
  mutate(across(where(is.numeric), ~ round(., 3)))

#Format table for results
kbl(rmse_s3, col.names = c("Method/Metric", "50", "100", "250",
                           "500", "1000", "2500", "1e+05"), booktabs = TRUE) %>%
    kable_styling(full_width = F, position = "left",
                  latex_options ="scale_down") %>%
    add_header_above(c(" " = 1, "Sample Size" = 7)) %>%
    row_spec(0, bold = TRUE) %>%
    kable_classic(full_width = F) %>%
    add_header_above(c("Study 3:" = 8), bold = TRUE, align = "l",
                     line = TRUE, italic = TRUE)
```

Study 3:

	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML_rmse	0.273	0.237	0.202	0.189	0.182	0.173	0.171
SEM_ULS_rmse	0.269	0.233	0.198	0.185	0.176	0.166	0.163
LSAM_ML_rmse	0.273	0.207	0.149	0.152	0.159	0.161	0.166
LSAM_ULS_rmse	0.272	0.207	0.149	0.152	0.159	0.161	0.166
GSAM_ML_rmse	0.274	0.207	0.149	0.152	0.159	0.161	0.166
GSAM_ULS_rmse	0.273	0.207	0.149	0.152	0.159	0.161	0.166

Across all estimators, the RMSE decreased with increasing N. For N=50 and N=1000-100000, none of the estimators differed from another. For N=100-500, the two SEM-estimators outperformed all SAM-estimators in terms of efficiency.

Thus, SEM outperformed SAM in smaller to moderate samples.

Summary Study 3

In terms of bias, all SAM estimators outperformed the two SEM estimators in smaller to moderate samples.

These results differed from the ones obtained with regards to SD and RMSE, where SEM tended to outperform SAM in smaller to moderate samples.

Importantly, one has to bear in mind that same as before, SAM only appears to be more robust against unmodelled positive residual correlations and cross-loadings due to its general negative bias in small samples. As in this study, same as in Robitzsch (2022), the misspecification values were positive, I again cannot conclude that SAM performed better than SEM.

Summary 2-factor-CFA models (Studies 1-3)

SAM did not generally outperform SEM in small to moderate samples. SAM exhibited a negative small sample bias that made SAM appear superior in conditions with unmodelled positive cross-loadings and residual correlations. This bias was especially strong for lower lambda and higher phi values. In the absence of misspecification or unmodelled negative cross-loadings and residual correlations, SAM tended to perform worse than traditional SEM.

Results Simulation 4: 5-factor-model under 3 different data-generating mechanisms

Importantly, for this study, I did not investigate CFA's anymore, but models with regressions included. Also, keep in mind that the naming of the DGM changed to correspond with the code implementation. What Robitzsch (2022) called DGM 1-3, I now named DGM 0-2.

Error, warnings and messages

```
#Save all errors, warnings and messages
all_messages_s4 <- readRDS("SimulationResults/sim4_results_error.rds")


#errors
errors_s4 <- all_messages_s4$errors
errors_sum_s4 <- unlist(errors_s4)
length(errors_sum_s4)
```

```
[1] 0
```

```
#warnings
warnings_s4 <- all_messages_s4$warnings
warnings_sum_s4 <- unlist(warnings_s4)
length(warnings_sum_s4)
```

```
[1] 9007
```

```
#messages
messages_s4 <- all_messages_s4$messages
messages_sum_s4 <- unlist(messages_s4)
length(messages_sum_s4)
```

```
[1] 0
```

No errors and messages were present. There were, however, 9007 warnings, I investigated in detail.

Warnings investigation

```
# Flatten the list of warnings into a single character vector
all_warnings_s4 <- unlist(warnings_s4)

# Count the occurrences of each unique warning
warn_s4 <- table(all_warnings_s4)

# Convert to data frame for better formatting
warn_s4 <- as.data.frame(warn_s4)
colnames(warn_s4) <- c("Warning Message", "Count")

warn_s4$`Warning Message` <- as.character(warn_s4$`Warning Message`)
warn_s4$`Count` <- as.character(warn_s4$`Count`)

# Define a helper function for conditional formatting
conditional_kable <- function(data) {
  if (knitr::is_latex_output()) {
    kable(data, booktabs = TRUE, longtable = TRUE
      , escape = FALSE, format = "latex" ) %>%
  kable_styling(latex_options = c("hold_position", "scale_down"),
```

```
                full_width = FALSE) %>%
  column_spec(1, width = "30em") %>%
  column_spec(2, width = "5em") %>%
  row_spec(0, bold = TRUE)
  } else {
    data
  }
}

conditional_kable(warn_s4)
```

```
Warning in styling_latex_scale(out, table_info, "down"): Longtable cannot be
resized.
```

Warning Message	Count
lavaan WARNING: covariance matrix of latent variables is not positive definite; use lavInspect(fit, ”cov.lv”) to investigate.	2
lavaan WARNING: number of observations (100) too small to compute Gamma	4500
lavaan WARNING: number of observations (50) too small to compute Gamma	4500
lavaan WARNING: Could not compute standard errors! The information matrix could not be inverted. This may be a symptom that the model is not identified.	4
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.595219e-17) is close to zero. This may be a symptom that the model is not identified.	1

There are three warnings that amount to 7 overall, that are referring to problems with regards to positive definite matrices or model identification. This suggests potential issues with multicollinearity or redundancy among latent variables, but only in a negligible number of estimations. The 9000 warnings with regards to Gamma computation in smaller samples are negligible as well, as I was not interested in the computation of robust fit indices.

Bias

```
# Load the absolute bias results
bias_ci_s4 <- readRDS("SimulationResultsProcessed/sim4_abs_bias_ci.rds")

#Change create_styled_table function names function for abs_bias
create_styled_table <- function(data, condition) {
  kbl(data,
      col.names = c("Method/Metric", "50", "100", "250",
                    "500", "1000", "2500", "1e+05"), booktabs = TRUE) %>%
    kable_styling(full_width = F, position = "left",
                  latex_options ="scale_down") %>%
    add_header_above(c(" " = 1, "Sample Size" = 7)) %>%
    row_spec(0, bold = TRUE) %>%
    kable_classic(full_width = F) %>%
    add_header_above(c("Condition:" = 8), bold = TRUE, align = "l",
                     line = TRUE, italic = TRUE) %>%
    footnote(general = paste("Condition:", condition))
}
```

Conditions without misspecification (DGM 0)

```
create_styled_table(bias_ci_s4[["DGM_0"]], "DGM_0")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.147	0.098	0.060	0.041	0.029	0.018	0.003
SEM_ULS	0.191	0.101	0.061	0.042	0.030	0.019	0.003
LSAM_ML	0.132	0.092	0.058	0.040	0.029	0.018	0.003

Note:

Condition: DGM_0

All estimators were biased for small to moderate samples (N=50-250). For higher N, no estimator was biased. In N=50, LSAM outperforms SEM-ULS. For all other conditions, there was no substantial difference between any of the estimators biases.

Thus, for the most part, no estimator generally outperformed another in the correctly specified model conditions.

Conditions with four unmodelled cross-loadings (DGM 1)

```
create_styled_table(bias_ci_s4[["DGM_1"]], "DGM_1")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.253	0.171	0.115	0.095	0.085	0.080	0.075
SEM_ULS	0.445	0.291	0.263	0.254	0.249	0.248	0.247
LSAM_ML	0.150	0.112	0.083	0.072	0.065	0.061	0.057

Note:

Condition: DGM_1

All estimators were biased across all conditions of N. Still, their bias decreased for higher N. LSAM-ML substantially outperformed the two SEM-estimators in smaller samples (N=50-100). It outperformed SEM-ULS in every other condition of N, as well. With regards to SEM-ML, LSAM-ML only slightly outperformed, for higher N.

Thus, SAM tended to outperform SEM in presence of unmodelled cross-loadings.

Conditions with 20 unmodelled residual correlation (DGM 2)

```
create_styled_table(bias_ci_s4[["DGM_2"]], "DGM_2")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.159	0.107	0.071	0.056	0.048	0.044	0.043
SEM_ULS	0.295	0.108	0.067	0.049	0.039	0.031	0.029
LSAM_ML	0.136	0.097	0.065	0.051	0.043	0.039	0.038

Note:

Condition: DGM_2

All estimators were biased for small to moderate samples (N=50-500). Only SEM-ULS was unbiased for N=500. For higher N (N=1000-100000), no estimator was biased. In N=50, LSAM outperformed SEM-ULS estimation. For all other conditions, there were no substantial differences between any of the estimators biases.

Summary

LSAM-estimation appeared to generally outperform SEM-ULS estimation in smaller samples across all DGM's. Moreover, LSAM- appeared to outperform both SEM-estimators in smaller samples, when unmodelled cross-loadings are present. Importantly, as I found LSAM to have a negative small sample bias in conditions with low values of phi as well in my simulation study 1b, we have to take this into account: Even though LSAM appeared to outperform SEM-estimation in DGM 1, this is attributable to its general negative small sample bias. Thus, it cannot be stated, that SAM estimation should be preferred in these scenarios.

In conditions without misspecification (DGM 0) and in the presence of unmodelled residual correlations (DGM 2), there was no general advantage of one estimation method over another, for smaller to moderate samples.

Comparison with original paper

Interestingly, unlike Robitzsch (2022), I have found some bias in DGM 0, even though the model was correctly specified in that scenario. For DMG 1, results are very similar, with

LSAM-ML seemingly outperforming SEM-ULS estimation. A new insight is that LSAM-ML seemed to outperform SEM-ML in smaller samples as well. In DGM 2, I did not find the disadvantage of SEM-ML in comparison to the other estimators that Robitzsch (2022) found. An additional finding was, that LSAM-ML tended to generally perform better than SEM-ULS in smaller samples. Importantly, my conclusion with regards to LSAM's performance was different, due to the different findings in my Study 1b, that found LSAM to be biased even for low values of phi and lambda.

RMSE

```
rmse_s4 <- readRDS("SimulationResults/sim4_rmse.rds")
```

Conditions without misspecification (DGM 0)

```
create_styled_table(rmse_s4[["DGM_0"]], "DGM_0")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.188	0.123	0.075	0.051	0.037	0.023	0.004
SEM_ULS	1.062	0.128	0.077	0.053	0.037	0.023	0.004
LSAM_ML	0.165	0.115	0.072	0.050	0.036	0.023	0.004

Note:

Condition: DGM_0

Across increasing N, the average RMSE decreased substantially for all estimators. For N=50, SEM-ULS performed substantially worse than the other two estimators. For all other conditions, there was no difference between any of the estimators.

Thus, no estimator generally outperformed another in conditions for a correctly specified model.

Conditions with four unmodelled cross-loadings (DGM 1)

```
create_styled_table(rmse_s4[["DGM_1"]], "DGM_1")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.373	0.257	0.166	0.124	0.107	0.100	0.095
SEM_ULS	2.070	0.373	0.320	0.306	0.300	0.298	0.296
LSAM_ML	0.188	0.141	0.103	0.089	0.080	0.075	0.071

Note:

Condition: DGM_1

Across increasing N, the average RMSE decreased substantially for all estimators. Across all N, SEM-ULS performed substantially worse than the other two estimators. For N=50-500, SEM-ML performed substantially worse than LSAM-ML. For higher N, SEM-ML performed worse than LSAM-ML as well, but only slightly.

Thus, LSAM-ML appeared to outperform the two SEM-estimators for small to moderate samples sizes, in the conditions with four unmodelled cross-loadings.

Conditions with 20 unmodelled residual correlation (DGM 2)

```
create_styled_table(rmse_s4[["DGM_2"]], "DGM_2")
```

Condition:							
	Sample Size						
Method/Metric	**50**	**100**	**250**	**500**	**1000**	**2500**	**1e+05**
SEM_ML	0.203	0.136	0.089	0.069	0.058	0.050	0.044
SEM_ULS	5.011	0.136	0.084	0.062	0.048	0.038	0.029
LSAM_ML	0.170	0.121	0.081	0.063	0.052	0.044	0.038

Note:

Condition: DGM_2

Across increasing N, the average RMSE decreased substantially for all estimators. For N=50, SEM-ULS performed substantially worse than the other two estimators. For all other conditions, there was no difference between any of the estimators.

Thus, no estimator generally outperformed another in conditions with 20 unmodelled residual correlation.

Summary

LSAM-ML appeared to outperform the two SEM-estimators in conditions with unmodelled cross-loadings. For conditions without misspecifications and unmodelled residual correlations, no difference arose besides a worse performance of SEM-ULS.

Comparison with original paper

The results align closely to the findings in Robitzsch (2022). A new insight is the comparatively worse performance of SEM-ULS in small samples (N=50).

Summary Study 4

Overall, the main performance difference is that LSAM appeared to be outperforming the two SEM-estimators in smaller to moderate samples sizes, in conditions with unmodelled cross-loadings (DGM1). This performance difference is attributed to its general negative small sample bias, that persisted even for low phi values in my Study 1b.

Thus, it cannot simply be stated that SAM estimation should be preferred in these scenarios. In conditions without misspecification (DGM 0) and in the presence of unmodelled residual correlations (DGM 2), there was no general advantage of one estimation method over another. A new insight was the comparatively worse performance of SEM-ULS in small samples (N=50). Due to my different findings with regards to the small sample bias of LSAM in Study 1b, I arrived at different conclusions than Robitzsch (2022).

Results Simulation 4a: 5-factor-model under 2 different data-generating mechanisms and varying beta size

Again, I investigated a five factor model with regressions included, and this time under varying size of the regression coefficient beta. Unlike the original paper, were the investigation was only conducted on the population level, I also varied the sample size as a factor in my simulation. Also, keep in mind that the naming of the DGM changed to correspond with the code implementation. What Robitzsch (2022) called DGM 1-3, I now called DGM 0-2.

Error, warnings and messages

```
#Save all errors, warnings and messages
all_messages_s4a <- readRDS("SimulationResults/sim4a_results_error.rds")


#errors
errors_s4a <- all_messages_s4a$errors
errors_sum_s4a <- unlist(errors_s4a)
length(errors_sum_s4a)
```

```
[1] 0
```

```
#warnings
warnings_s4a <- all_messages_s4a$warnings
warnings_sum_s4a <- unlist(warnings_s4a)
length(warnings_sum_s4a)
```

```
[1] 24089
```

```
#messages
messages_s4a <- all_messages_s4a$messages
messages_sum_s4a <- unlist(messages_s4a)
length(messages_sum_s4a)
```

```
[1] 0
```

No errors and messages were present. There were, however, 24089 warnings, I investigated in detail.

Warnings investigation

```
# Flatten the list of warnings into a single character vector
all_warnings_s4a <- unlist(warnings_s4a)

# Count the occurrences of each unique warning
warn_s4a <- table(all_warnings_s4a)

# Convert to data frame for better formatting
warn_s4a <- as.data.frame(warn_s4a)
colnames(warn_s4a) <- c("Warning Message", "Count")

warn_s4a$`Warning Message` <- as.character(warn_s4a$`Warning Message`)
warn_s4a$`Count` <- as.character(warn_s4a$`Count`)

#Using formatting function from previous study
conditional_kable(warn_s4a)
```

```
Warning in styling_latex_scale(out, table_info, "down"): Longtable cannot be
resized.
```

Warning Message	Count
lavaan WARNING: covariance matrix of latent variables is not positive definite; use lavInspect(fit, ”cov.lv”) to investigate.	4
lavaan WARNING: number of observations (100) too small to compute Gamma	12000
lavaan WARNING: number of observations (50) too small to compute Gamma	12000
lavaan WARNING: Could not compute standard errors! The information matrix could not be inverted. This may be a symptom that the model is not identified.	19
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -1.034416e-12) is smaller than zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -1.569249e-17) is smaller than zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -1.845280e-05) is smaller than zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -2.494609e-18) is smaller than zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -2.991214e-12) is smaller than zero. This may be a symptom that the model is not identified.	1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -4.836987e-10) is smaller than zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -5.183240e-12) is smaller than zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -5.404489e-19) is smaller than zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -7.959258e-05) is smaller than zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= -8.553561e-18) is smaller than zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.061975e-17) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.069564e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.082677e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.140712e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.176210e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.263219e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.267379e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.269706e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.280797e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.304550e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.307111e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.341988e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.348520e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.467929e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.562208e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.595219e-17) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.599418e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.613468e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.663466e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.719905e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 1.725270e-12) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.111130e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.112887e-17) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.156102e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.258900e-15) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.538499e-15) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.643121e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.802886e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 2.989425e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 3.548189e-15) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 3.582932e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 3.595421e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 3.884589e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 4.582696e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 4.775365e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 5.010298e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 5.058328e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 5.160987e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 5.467288e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 5.818820e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 6.048167e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 6.302792e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 6.541549e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 6.596545e-18) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 6.696249e-14) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 7.237797e-15) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 7.470915e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 7.652385e-16) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 7.813113e-13) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 8.082302e-15) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 8.432601e-17) is close to zero. This may be a symptom that the model is not identified. 1

lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 8.443523e-13) is close to zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 8.502271e-13) is close to zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 9.003130e-14) is close to zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 9.086626e-13) is close to zero. This may be a symptom that the model is not identified.	1
lavaan WARNING: The variance-covariance matrix of the estimated parameters (vcov) does not appear to be positive definite! The smallest eigenvalue (= 9.923178e-14) is close to zero. This may be a symptom that the model is not identified.	1

The 24000 warnings with regards to Gamma computation in smaller samples were negligible as well, as I was not interested in the computation of robust fit indices. All 89 other warnings referred to problems with positive definite matrices or model identification. This suggests potential issues with multicollinearity or redundancy among latent variables, but only in a negligible number of estimations.

Bias

```
# Load the absolute bias results
bias_ci_s4a <- readRDS("SimulationResultsProcessed/sim4a_abs_bias_ci.rds")
```

Conditions with four unmodelled cross-loadings (DGM 1)

```
#Normal print
#print(bias_ci_s4a[["DGM_1"]][["SEM_ML_metrics"]])
#print(bias_ci_s4a[["DGM_1"]][["SEM_ULS_metrics"]])
#print(bias_ci_s4a[["DGM_1"]][["LSAM_ML_metrics"]])

#Formatted tables
kbl(bias_ci_s4a[["DGM_1"]][["SEM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of SEM-ML for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of SEM-ML for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.253	0.172	0.115	0.095	0.085	0.080	0.075
0.2	0.327	0.220	0.162	0.139	0.122	0.114	0.109
0.3	0.545	0.270	0.218	0.206	0.195	0.190	0.180
0.4	0.981	0.336	0.270	0.257	0.252	0.251	0.254

```
kbl(bias_ci_s4a[["DGM_1"]][["SEM_ULS_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
```

```
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of SEM-ULS for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of SEM-ULS for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.445	0.291	0.263	0.254	0.249	0.248	0.247
0.2	0.697	0.324	0.285	0.276	0.270	0.269	0.267
0.3	1.079	0.429	0.322	0.308	0.299	0.298	0.294
0.4	1.991	0.654	0.385	0.357	0.344	0.339	0.333

```
kbl(bias_ci_s4a[["DGM_1"]][["LSAM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of LSAM-ML for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of LSAM-ML for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.150	0.112	0.083	0.072	0.065	0.061	0.057
0.2	0.145	0.108	0.077	0.066	0.059	0.056	0.053
0.3	0.141	0.104	0.073	0.060	0.052	0.047	0.043
0.4	0.151	0.115	0.088	0.076	0.070	0.067	0.065

All three estimators were biased across all conditions. The only exception is LSAM for very large samples (N=2500-100.000) and beta=0.3. For all three estimators, across increasing N, the bias decreased substantially, in all conditions of beta. For higher beta, the bias increased substantially, in all conditions of N for SEM-ML and SEM-ULS. This was not the case for LSAM-ML.

Across all conditions, LSAM-ML substantially outperformed the two SEM-estimators. The bias difference between LSAM-ML and the two other estimators was higher for lower N. The only exception was for beta =0.1 in N=250 and all higher N. Here, there was only a slightly better performance of LSAM- over SEM-ML estimation, though not substantial. This effect was stronger for higher values of beta. Additionally, SEM-ML outperformed SEM-ULS in all conditions as well.

Thus, LSAM-ML generally appeared to outperform SEM-estimation and SEM-ML outperformed SEM-ULS estimation, in the presence of four unmodelled cross-loadings. This was especially evident for higher beta values and lower N.

Conditions with 20 unmodelled residual correlation (DGM 2)

```
#Normal print
#print(bias_ci_s4a[["DGM_2"]][["SEM_ML_metrics"]])
#print(bias_ci_s4a[["DGM_2"]][["SEM_ULS_metrics"]])
#print(bias_ci_s4a[["DGM_2"]][["LSAM_ML_metrics"]])


#Formatted tables
```

```
kbl(bias_ci_s4a[["DGM_2"]][["SEM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of SEM-ML for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of SEM-ML for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.159	0.107	0.071	0.056	0.048	0.044	0.043
0.2	0.168	0.112	0.072	0.056	0.046	0.041	0.038
0.3	0.181	0.122	0.080	0.063	0.052	0.046	0.041
0.4	0.218	0.145	0.098	0.080	0.069	0.063	0.055

```
kbl(bias_ci_s4a[["DGM_2"]][["SEM_ULS_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of SEM-ULS for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of SEM-ULS for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.295	0.108	0.067	0.049	0.039	0.031	0.029
0.2	0.257	0.113	0.070	0.052	0.040	0.032	0.027
0.3	0.353	0.125	0.078	0.059	0.047	0.038	0.032
0.4	0.330	0.157	0.096	0.075	0.063	0.055	0.047

```
kbl(bias_ci_s4a[["DGM_2"]][["LSAM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left",
                latex_options ="scale_down") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: Absolute bias of LSAM-ML for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: Absolute bias of LSAM-ML for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.136	0.097	0.065	0.051	0.043	0.039	0.038
0.2	0.130	0.091	0.058	0.043	0.033	0.025	0.019
0.3	0.125	0.088	0.057	0.042	0.033	0.026	0.022
0.4	0.133	0.100	0.077	0.068	0.064	0.061	0.057

All estimators were generally biased across conditions, besides conditions of very high N (N=1000-100.000) and lower to moderate beta (0.1-0.3). For all three estimators, across increasing N, the bias decreased substantially, in all conditions of beta. Only in SEM-ML

in N=50 and SEM-ULS in N=100, there was a substantial increase in bias for higher beta values.

For N=50, both SEM- and LSAM-ML substantially outperformed ULS estimation. In these conditions, LSAM-ML also outperformed SEM-ML. for beta 0.2-0.4. Lastly, LSAM-ML outperformed SEM-ULS in N=100 for beta = 0.4.

For all other conditions, none of the estimators differed from another, in terms of bias.

Thus, LSAM-ML appeared to perform best in small samples and higher beta. Besides that, none of the estimators generally outperformed another in the presence of 20 unmodelled residual correlations.

RMSE

```
# Load the rmse results
rmse_s4a <- readRDS("SimulationResults/sim4a_rmse.rds")
```

Conditions with four unmodelled cross-loadings (DGM 1)

```
#Normal print
#print(rmse_s4a[["DGM_1"]][["SEM_ML_metrics"]])
#print(rmse_s4a[["DGM_1"]][["SEM_ULS_metrics"]])
#print(rmse_s4a[["DGM_1"]][["LSAM_ML_metrics"]])

#Formatted tables
kbl(rmse_s4a[["DGM_1"]][["SEM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
```

```
  add_header_above(c("Study 4a: RMSE of SEM-ML for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of SEM-ML for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.451	0.423	0.403	0.394	0.391	0.390	0.390
0.2	0.573	0.532	0.516	0.505	0.498	0.493	0.492
0.3	0.701	0.609	0.595	0.591	0.586	0.584	0.577
0.4	0.949	0.674	0.648	0.644	0.643	0.643	0.645

```
kbl(rmse_s4a[["DGM_1"]][["SEM_ULS_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: RMSE of SEM-ULS for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of SEM-ULS for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.558	0.515	0.506	0.503	0.502	0.502	0.502
0.2	0.771	0.594	0.577	0.573	0.571	0.570	0.570
0.3	0.924	0.688	0.634	0.629	0.625	0.624	0.623
0.4	1.221	0.816	0.690	0.678	0.673	0.671	0.669

```
kbl(rmse_s4a[["DGM_1"]][["LSAM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: RMSE of LSAM-ML for DGM 1" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of LSAM-ML for DGM 1

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.358	0.360	0.365	0.366	0.367	0.367	0.368
0.2	0.425	0.429	0.433	0.435	0.435	0.435	0.436
0.3	0.442	0.449	0.454	0.455	0.455	0.456	0.456
0.4	0.417	0.425	0.429	0.430	0.431	0.431	0.431

Across all conditions of N and beta, LSAM-ML estimation was more efficient than the two SEM-estimators. This difference was stronger in smaller samples. Additionally, SEM-ML outperformed SEM-ULS estimation, in terms of efficiency, in all conditions as well. Both SEM estimators had reduced RMSE values for higher N. All three estimators had higher RMSE values for higher beta. This effect was especially present in the two SEM estimators.

Thus, LSAM-ML appeared to outperform SEM-estimation and SEM-ML appeared to outperform SEM-ULS estimation, in the presence of four unmodelled cross-loadings. This was especially true in lower samples and higher beta values.

Conditions with 20 unmodelled residual correlation (DGM 2)

```
#Normal print
#print(rmse_s4a[["DGM_2"]][["SEM_ML_metrics"]])
```

```
#print(rmse_s4a[["DGM_2"]][["SEM_ULS_metrics"]])
#print(rmse_s4a[["DGM_2"]][["LSAM_ML_metrics"]])

#Formatted tables
kbl(rmse_s4a[["DGM_2"]][["SEM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: RMSE of SEM-ML for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of SEM-ML for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.378	0.370	0.367	0.366	0.365	0.365	0.365
0.2	0.464	0.453	0.448	0.447	0.446	0.445	0.445
0.3	0.524	0.512	0.505	0.503	0.502	0.502	0.501
0.4	0.577	0.557	0.548	0.545	0.543	0.543	0.542

```
kbl(rmse_s4a[["DGM_2"]][["SEM_ULS_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: RMSE of SEM-ULS for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of SEM-ULS for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.377	0.352	0.347	0.346	0.345	0.344	0.344
0.2	0.452	0.443	0.437	0.435	0.433	0.433	0.433
0.3	0.616	0.506	0.497	0.494	0.493	0.492	0.492
0.4	0.626	0.553	0.542	0.539	0.536	0.536	0.535

```
kbl(rmse_s4a[["DGM_2"]][["LSAM_ML_metrics"]],
    col.names = c("beta", "N=50", "N=100", "N=250", "N=500",
                  "N=1000", "N=2500", "N=100.000"), booktabs = TRUE) %>%
  kable_styling(full_width = F, position = "left") %>%
  add_header_above(c(" " = 1, "Sample size" = 7)) %>%
  row_spec(0, bold = TRUE) %>%
  kable_classic(full_width = F) %>%
  add_header_above(c("Study 4a: RMSE of LSAM-ML for DGM 2" = 8),
                   bold = TRUE, align = "l", line = TRUE, italic = TRUE)
```

Study 4a: RMSE of LSAM-ML for DGM 2

	Sample size						
beta	**N=50**	**N=100**	**N=250**	**N=500**	**N=1000**	**N=2500**	**N=100.000**
0.1	0.352	0.354	0.356	0.357	0.357	0.357	0.357
0.2	0.417	0.420	0.423	0.423	0.423	0.423	0.424
0.3	0.439	0.443	0.446	0.447	0.447	0.447	0.447
0.4	0.419	0.424	0.427	0.428	0.428	0.428	0.429

Across all conditions of N, LSAM-ML generally outperformed the two SEM-estimators for lower to moderate beta values (approx. beta= 0.2-0.4). This effect was stronger for higher values of beta and lower N. SEM-ULS had reduced RMSE values for higher N. All three estimators had higher RMSE values for higher beta. This effect was especially present in the two SEM estimators.

Thus, LSAM-ML appeared to generally outperform SEM-estimation, in the presence of 20 unmodelled residual correlation. Even more so for higher values of beta and lower N.

Summary Study 4a

In terms of bias, only in the presence of four unmodelled cross-loadings, did LSAM-estimation appear to generally outperform both SEM-estimators. This was especially evident for higher beta values and lower N. In the presence of unmodelled residual correlations, LSAM-ML appeared to perform best in small samples and higher beta as well.

With regards to RMSE, LSAM-ML appeared to outperform SEM-estimation both under unmodelled cross-loadings and residual correlations. Again, this appeared to be especially present in conditions with small samples and high beta values.

Going all the way back to Studies 1-3, we again have to remember that LSAM exhibited a general negative bias in smaller samples for the 2-factor-CFA. This made LSAM appear to outperform in conditions of lower N and higher phi or beta. As replicated in Study 1b, this bias tended to be even stronger for higher parameter values in the structural model, and thus for higher beta values as well.

Comparison to original paper

Importantly, Robitzsch (2022) did not investigate differential effects due to sample size. Nor did they investigate RMSE. Thus, results are only partly comparable.

In DMG1 (which was called DGM 2 in the original paper), the original paper indicated the same results, in that LSAM performed best. Interestingly, Robitzsch (2022) did not find the better performance of SEM-ML over SEM-ULS estimation that I found. Also, no differential effect for higher beta values was found in the original paper.

In DGM2, Robitzsch (2022) found SEM-ULS to perform slightly better than the two others. Thus, results are quite similar, as I also did not find substantial differences between the three estimators.

I disagree with the conclusion that Robitzsch (2022) drew with regards to this study, as they deemed SEM-ULS and LSAM do not differ in performance, when in fact LSAM outperformed SEM-ULS in DGM1 in their studies.

Additionally, later on in the paper, they posited that SAM can be expected to be more robust than both SEM-estimators in models with non-saturated structural models. With this last conclusion, I disagree with regards to the now multiple times mentioned potential negative small sample bias of LSAM, that was present for lower phi/ beta values as well.

Summary for 5-factor-model under 3 different data-generating mechanisms (Studies 4 and 4a)

For low beta (0.1), LSAM appeared to outperform the two SEM-estimators in smaller to moderate samples sizes, in conditions with unmodelled cross-loadings (DGM1). In conditions without misspecification (DGM 0) and in the presence of unmodelled residual correlations (DGM 2), there was no general advantage of one estimation method over another.

With increasing values of beta (0.2-0.4), the effect with regards to unmodelled cross-loadings became even stronger (DGM 1). In this context, LSAM appeared to not only outperform SEM-estimation in terms of efficiency in small to moderate samples in DGM1, but also in conditions with unmodelled residual correlations (DGM2).

Very important to keep in mind here is the importang finding of the earlier study and the original paper, that LSAM exhibits a negative finite sample bias that skewed results in conditions with positive values for unmodelled cross-loadings or residual correlations (Robitzsch 2022). This effect was even stronger for higher values of phi and low N, which aligns with the findings in my studies for models with regressions.

Overall summary

With regards to all studies conducted, as in the original paper by Robitzsch (2022), SAM did not generally outperform SEM in small to moderate samples. SAM exhibited a negative small sample bias that made SAM appear superior in conditions with unmodelled positive cross-loadings and residual correlations. This bias was especially strong for lower lambda and higher phi or beta values. Going ahead of what was investigated in Robitzsch (2022), I found that this bias is also present in models with lower phi or beta values. Thus, it cannot be concluded that SAM is more robust in models with non-saturated structural parameters. If

there was no misspecification or unmodelled negative cross-loadings and residual correlations, SAM tended to perform worse than traditional SEM, as far as can be concluded from the 2-factor-CFA-models.

Bibliography

Robitzsch, Alexander. 2022. "Comparing the Robustness of the Structural After Measurement (SAM) Approach to Structural Equation Modeling (SEM) Against Local Model Misspecifications with Alternative Estimation Approaches." *Stats* 5 (3): 631–72. https://doi.org/10.3390/stats5030039.

Rosseel, Yves, and Wen Wei Loh. 2022. "A Structural After Measurement Approach to Structural Equation Modeling." *Psychological Methods*, No Pagination Specified–. https://doi.org/10.1037/met0000503.