MITNB Workshop: Estimating internal consistency reliability for intensive longitudinal data: A brief tutorial - Practical guide

Sebastian Castro-Alvarez

2025-04-11

This practical guide was developed for the workshop *Estimating internal consistency reliability for intensive longitudinal data: A brief tutorial* offered on November 4th during the MITNB meeting 2025. To perform the analyses to estimate the internal consistency reliability based on the different approaches, you will use R (R Core Team, 2025), and, optionally, Mplus (version 8.0 or higher, Muthén & Muthén, 1998-2025). To follow this guide, basic R skills are required. You are expected to be familiar with loading R packages, importing and manipulating data, and using functions for your statistical analyses with aid of the help pages. Here, I do not give the exact answer but instead I give rough instructions and indications to point you to the right direction.

> **Note**
>
> You are expected to create a new R script from scratch, where you will load the required R packages, the data you are going to work with, and write the commands to estimate the reliability while following this guide.

Materials for this workshop are openly available in the GitHub repository ReliabilityWorkshop-MITNB2025. In this repository, you will find this practical guide, the slides used during the workshop, and the file *2RDM_sim.R* to simulate data. You will also find the following folders:

- **Data/**: Here you find the simulated data you can work with.
- **Mplus/**: Example Mplus syntaxes for the 2RDM and ME-TSO models.
- **R**: Custom R functions developed for this workshop.
- **Solutions**: R scripts and Mplus input files to estimate the reliability of the two different data sets based on the different approaches.

> **Tip**
>
> I recommend that you download the whole repository into your local drive. One way to do this is to click on the green button "Code" and select "Download ZIP".

From this point onwards, when referencing to specific files, I use the relative paths within the folder structure found on GitHub, so it is easy for you to find them.

Learning Goals

The goal of this guide is that you learn how to estimate the internal consistency reliability of intensive longitudinal data according to the six/seven approaches discussed during this workshop. More specifically, the learning goals are:

- Estimate the within- and between-person reliability of ESM data based on the generalizability theory (Cranford et al., 2006), multilevel modeling (Nezlek, 2017), and the multilevel confirmatory factor model Geldhof et al. (2014).
- Estimate person-specific reliability coefficients of ESM data according to the p-technique factor analysis Hu et al. (2016) and the dynamic factor model Fuller-Tyszkiewicz et al. (2017).
- Estimate the between-person and person-specific reliability coefficients of ESM data according to the 2RDM (Xiao et al., 2023).
- Estimate person and item-specific reliability coefficients of ESM data based on the ME-TSO model (Castro-Alvarez et al., 2022).

Setup

In addition to R and your preferred R interface, you will need the following R packages and some custom R functions to complete the reliability analysis with the different approaches. Also, for the 2RDM and ME-TSO, Mplus is needed, however this is optional not only because access to Mplus is limited but also because these approaches are the more time-consuming and there won't be enough time during the workshop to run these analyses. The R packages needed are:

- psych (William Revelle, 2025) for the computation of the reliability based on the generalizability theory and multilevel modeling.
- misty (Yanagida, 2025) for the computation of the reliability based on the multilevel confirmatory factor model.
- lavaan (Rosseel, 2012) for the estimation of the p-technique factor analysis.
- dynr (Ou et al., 2019) for the estimation of the dynamic factor model. Installing this package is not trivial, careful follow the installation instructions found here.

- MplusAutomation (Hallquist & Wiley, 2018) to read into R the output generated from Mplus when running the 2RDM or the ME-TSO.

You will also find the following R files with custom functions helpful for some approaches:

- *R/aux_pfa.R* includes four functions that aid on fitting the p-technique and dynamic factor model to each individual in a ESM data set.
- *R/aux_mplus.R* includes six functions useful to read the output of the ME-TSO and the 2RDM into R and to compute the corresponding reliability coefficients based on the Mplus output.

> Tip
>
> You can load these functions into your R environment with the `source()` function: `source("R/aux_pfa.R")`

Data

For this practical guide, you need an intensive longitudinal data set in which a set of items (e.g., enthusiastic, energetic, active) measuring *one* construct (e.g., positive affect) were repeatedly used to measure a sample of individuals. On top of the responses to each item, the data set must also include an *id* variable and a *time* variable. The *time* variable should be a numeric variable that indicates, for example, the occasion number or the day number, which depends on the study design. You have the option to work with one of the following three data sets:

- **Own data**: If you have data from a previous or current project available that satisfies the previous requirements, feel free to use it for this workshop. The analyses that you will perform will probably be useful for your own research.
- **Data/data2rdm.dat**: If you want to work with a data set for which all approaches will work just fine, you can use this simulated data set. The data was simulated based on the 2RDM and includes the responses on 6 items of 100 individuals, each with 90 repeated measures. In this data set, you will find the *id* variable; the *time* variable, which can be thought as day number; and the variables *I1–I6* with the responses to the six items. The solution to estimating the reliability according to all the approaches can be found in the R script *Solutions/simdata/solution_simdata.R*. If you feel like changing the number of items, persons, or timepoints or you want to adjust the true parameters to your preference, you can generate a new data set by modifying the R script *2RDM_sim.R*.
- **ELISA_CR.rds**: If you feel daring and up for a challenge, try using this data set from Vogelsmeier et al. (2024). You can download these data from this OSF repository. In summary, this is an empirical ESM data from 71 individuals. The participants in this study filled in a 7-point Likert scale 10-item questionnaire about momentary life satisfaction

up to six times a day during 14 days for a maximum of 84 measurements. On average, participants completed 61.6 measurements with a minimum of 22 and a maximum of 84. The data includes 24 variables but for the analyses in this guide you only need the variables *id*, *n_obs*, and the 10 items: *consider_happy*, *satisfied_life*, *change_many*, *life_ideal*, *makes_happy*, *satisfied_situation*, *change_nothing*, *leaves_desired*, *important_things*, and *line_life*. One particularity of this study is that it has a planned missingness design in which participants did not have to report on all 10 items on every occasion. Instead, the item *line_life* was an anchor item, which was always shown to the participants and another six items of the remaining nine were randomly selected for each occasion. You can find the detailed solution to estimating the reliability according to all the approaches in the R script *Solutions/ELISA_CR/solution_elisa_cr.R*.

Computing the Internal Consistency Reliability

Generalizability Theory and Multilevel Modeling

Both, the generalizability theory and the multilevel approach allow estimating a between- and a within-person reliability coefficient. In both cases, the procedure consists of fitting a mixed-effects model, extracting the estimated variance components, and using the latter to compute the corresponding reliability coefficients. An easy way to do this in R is with the function `mlr()` or `multilevel.reliability()` from the `psych` package. To use this function, your data can be in the traditional long format, in which each row has the responses to the items of one individual at a given time point; or your data can be in a longer format (*"hyperlong"*), in which each row has the response to one item of one person at a given time point. In the *hyperlong* format, your data should have four variables: *id*, *time*, *item indicator*, and *item response*. Try to use this function with the data of your choice.

> **Tip**
>
> Read carefully the function options as some of these can make it very time consuming. Generally, you would want `lmer = TRUE`, and `icc = FALSE`.

> **Warning**
>
> If you have a very large data set (more than 200 participants and each with more than 100 time points), running the `mlr` function can take several minutes even more than an hour. In this case, it is best to exclusively request the reliability coefficients based on the multilevel modeling approach. This can be done by setting `lmer = FALSE` and `lme = TRUE`.

The output of the `mlr()` automatically gives you the estimated reliability coefficients based on the two approaches. For the generalizability theory, the function retrieves the *RkF*, *R1R*, *RkR*,

and *Rc*. *RkF* or *RkR* is the between-person reliability depending on your study design, and *Rc* is the within-person reliability. For the multilevel modeling approach, the function retrieves the *RkRn* and *Rcn*, which are the between- and within-person reliability coefficients, respectively.

> **Tip**
>
> Alternatively, you can directly use the function `lmer` of the `lme4` package to compute the reliability coefficients on your own. For an example of this, check the supplementary materials of Castro-Alvarez et al. (2025).

Multilevel Confirmatory Factor Analysis

The estimation of the reliability based on multilevel confirmatory analysis suggested by Geldhof et al. (2014) and later revisited by Lai (2021) consists of fitting the model to the data and using the estimated factor loadings and measurement error variances to compute the between- and within-person reliability coefficients. Fortunately, the procedure to compute these coefficients as suggested by Lai (2021) has been implemented in the function `multilevel.omega()` from the `misty` package. This function uses `lavaan` in the background to fit the model and automatically computes the reliability coefficients of interest with their corresponding confidence intervals. For this function you need your data in the traditional long format. Try to use this function with the data of your choice.

> **Tip**
>
> When Lai (2021) revisited the multilevel confirmatory factor analysis reliability, he also accounted for the types of constructs that can be found in multilevel models (i.e., within-person, shared, and configural). Which reliability coefficients are estimated depend on the type of construct your are working with. Generally, in ESM data, multilevel constructs are configural, so you should use the option `const = "config"` in `multilevel.omega()`.

The function `multilevel.omega()` reports three reliability coefficients, namely, the within-person reliability, ω^w, the between-person reliability, ω^b, and the composite reliability, ω^{2l}. The first two are the more important and have a similar meaning to the reliability coefficients as estimated based on the two previous approaches.

> **Tip**
>
> If you obtain an improper solution, meaning you get warnings messages about negative variances or singular matrices, consider using the option `fix.resid = "all"`. This fixes the residual variances at the between-level at 0 and might help to obtain proper and interpretable results. However, beware that doing this implies that you assume that

strong factorial invariance holds.

P-technique Factor Analysis

When using the p-technique factor analysis, you are fitting a factor model to the responses of each person. Due to this, it might not be possible to use this approach on all participants if a participant did not complete enough measurement occasions. As a rule of thumb, this approach should not be used when a participant has less than 50 observations. Yet, this also depends on the number of planned observations per participant.

Overall, the procedure to compute the person-specific reliability coefficients for each participant in your data consists of the following steps:

1. Get the responses to the items of a given participant.
2. Fit an unidimensional confirmatory factor model with `lavaan` to the participant's data. To identify this model, you can either fix the variance of the latent factor at 0 or the factor loading of the first item at 1.
3. If the model did not converge or resulted in an improper solution, go back to step 2 and try a different model either by making it multidimensional or by excluding some items.
4. If the model converged, extract the estimated parameters and compute the person-specific reliability based on the following formula:

$$\omega_i = \frac{(\sum_{j=1}^{M} \lambda_j)^2 \sigma_\zeta^2}{(\sum_{j=1}^{M} \lambda_j)^2 \sigma_\zeta^2 + \sum_{j=1}^{M} \sigma_{v_j}^2}, \tag{1}$$

   where, λ_j is the factor loading for the j item, σ_ζ^2 is the variance of the latent factor, and $\sigma_{v_j}^2$ is the measurement error variance of the j item.

> **Tip**
>
> When you only have three items measuring the construct of interest, further model constraints are needed for the model to be identified. In such cases, all factor loadings have to be fixed at 1.

> **Tip**
>
> Detailed code to obtain the reliability coefficients based on p-technique factor analysis can be found in the supplementary materials of Castro-Alvarez et al. (2025).

For this workshop, I have written the function `esm_pfa_omega()` that automatically fits an unididimensional factor model to each person and computes its corresponding reliability

coefficient. For the participants for whom the model did not converge or had an improper solution, no reliability coefficient is computed. This function uses the following arguments:

- `data`: a data.frame of the data in traditional long format.
- `idlab`: a character with the name of the id variable.
- `items`: a character vector with the name of the items.
- `constraint`: either "variance", "loading" or "tau". Defines how the cfa model is identified. If "variance", the variance of the latent factor is fixed at 1. If "loading", the factor loading of the first item is fixed at 1. If "tau", all factor loadings are fixed at 1, which is useful when there are 3 or less items. Default is "variance"
- `missing`: options to handle missing values available in lavaan. Recommended options are "listwise" or "fiml". Default is "listwise".
- `...`: additional arguments available in `cfa()`.

> Tip
>
> If the function fails to run, it might be because your data is not a data.frame.

This function returns a list, where the first element is a data.frame that includes the persons' id, the estimated person-specific reliability coefficients, and a TRUE/FALSE variable indicating if the model fits the data well according to CFI, TLI, and RMSEA.

You can either try to do the whole procedure by your own or use the custom function `esm_pfa_omega()`.

Dynamic Factor Analysis

Similar to the p-technique, the dynamic factor analysis approach implies fitting the model to each participant. The difference is that the dynamic factor model includes lag effects on the latent variable. Software packages where you can specify lagged latent variables are for example `Mplus`, `OpenMx`, or `dynr`. For these analyses, also consider including participants with at least 50 measurement occasions.

The procedure to compute the person-specific reliability coefficients for each participant based on the dynamic factor model in `dynr` consists of the following steps:

1. Get the responses to the items of a given participant.
2. Define the starting values for the model. You can define them on your own or use the estimated parameters from fitting a p-technique factor analysis as starting values.
3. Define the "recipes" of the model in `dynr` with the functions: `prep.matrixDynamics()`, `prep.measurement()`, `prep.noise()`, and `prep.initial()`.
4. Define the `dynr` model with the function `dynr.model()`.
5. Fit the model to the data with the function `dynr.cook`.

6. If the model converged (`exitflag` equal to 1, 2, or 3), extract the estimated parameters and compute the person-specific reliability based on Equation Equation 1 but with

$$\sigma_\zeta^2 = \frac{\sigma_\varepsilon^2}{1-\phi^2},$$

where, σ_ε^2 is the residual variance of the latent autoregressive process, and ϕ is the estimated lag-1 autoregressive effect.

> **Tip**
>
> Detailed code to obtain the reliability coefficients based on dynamic factor analysis can be found in the supplementary materials of Castro-Alvarez et al. (2025).

For this approach, I wrote the function `esm_dfa_omega()` that automatically fits an unidimensional lag-1 factor model to each person and computes its corresponding reliability coefficient. Note that no reliability coefficient is computed when the model for a person does not converge. This function uses the following arguments:

- `data`: a data.frame of the data in traditional long format.
- `idlab`: a character with the name of the id variable.
- `timelab`: a character with the name of the time variable.
- `items`: a character vector with the name of the items.
- `...`: additional arguments available in `cfa()`, which is used to create starting values.

> **Tip**
>
> If the function fails to run, it might be because your data is not a data.frame.

This function returns a list, where the first element is a data.frame that includes the persons' id, the estimated person-specific reliability coefficients, and the estimated autoregressive effect.

You can either try to do the whole procedure by your own or use the custom function `esm_dfa_omega()`.

2RDM

> **Note**
>
> For the estimation of the 2RDM and the ME-TSO approaches, Mplus is needed. If you do not have access to Mplus, you have finished this practical guide.

The 2RDM is a multilevel dynamic factor model, which allows estimating the between-person reliability coefficient and person-specific reliability coefficients of the sample. This model was

originally proposed to analyze multiple constructs simultaneously, however, in this practical, we will use a simplified version for the analysis of one unidimensional construct.

> **Note**
>
> Leertouwer et al. (2025) have implemented the 2RDM in JAGS giving a foss alternative to estimate the model.

In short, to use this approach, you need to fit the model in Mplus, while saving the MCMC samples and plausible values of the factor scores, and use these to compute all the reliability coefficients of interest. However, doing the latter can be quite cumbersome. For this reason, I wrote a custom function that automatizes this process and will make everything easier for you.

> **Tip**
>
> As this model is estimated within the Bayesian framework, when computing new quantities such as the reliability coefficients, one can obtain the posterior distribution of such new quantities by repeatedly computing them using the plausible values of the parameters and random effects across iterations.

To fit the 2RMD and compute the corresponding reliability coefficients, you can follow these steps:

1. Export your data to Mplus with help of the function `prepareMplusData()` from the package `MplusAutomation`.

> **Tip**
>
> Before exporting your data to Mplus, make sure that your variable names are maximum 8 characters long and that your `id` variable is numeric. Ideally, the label of your `id` variable is also `id`, but this is not entirely necessary.

> **Tip**
>
> `prepareMplusData()` can also create an initial Mplus input file where you can write your model.

2. Write the 2RDM model syntax for your variables. To do this, you can use the file *"Mplus/ex_2rdm.inp"* as a template.

> Tip
>
> Make sure that you save the MCMC samples and plausible values of the factor scores with the **SAVEDATA** command and that you request the **TECH8** output.

3. Run the model directly in Mplus or from within R with the function `runModels()` from the package `MplusAutomation`. The advantage of running the model directly in Mplus is that it prints the progress. This is helpful to adjust your expectations on how long it will take to run the model. In contrast, this is not possible when running the model from within R.
4. Verify that your model converged by checking in the output file that the maximum *PSR* value is lower than 1.1 and relatively stable for the last 1000 iterations.
5. Use the custom function `reliability_2rdm()` to read the output into R and automatically compute all the reliability coefficients of interest. This function takes as input the name of the Mplus output file and a vector with the names of the items. Other arguments are explained in the source code.

> Tip
>
> Depending on the size of your data, the file where the MCMC of the plausible values are stored can be quite big, taking a couple of GB of storage. Because of this, the function `reliability_2rdm()` can be quite slow as it has to read such a big file into R.

Using the function `reliability_2rdm()` will return a list with four elements. The first one are the estimates of the person-specific reliability coefficients, the second one is the estimate of the between-person reliability coefficients, and the other two include the corresponding MCMC samples.

ME-TSO

The ME-TSO is also a multilevel dynamic factor model, but encompassed within the latent state-trait theory. As a consequence, the procedure to compute the reliability coefficient is quite different from the other approaches as it is done per item and person. Similarly to the 2RDM, one needs to fit the model and use the MCMC samples of the parameters and plausible values to compute the corresponding coefficients. For which, I have also written a custom function to facilitate this task.

You can follow these steps to estimate the reliability based on the ME-TSO:

1. Export your data to Mplus with help of the function `prepareMplusData()` from the package `MplusAutomation`.

> **Tip**
>
> Before exporting your data to Mplus, make sure that your variable names are maximum 8 characters long and that your `id` variable is numeric. Ideally, the label of your `id` variable is also `id`, but this is not entirely necessary.

> **Tip**
>
> `prepareMplusData()` can also create an initial Mplus input file where you can write your model.

2. Write the ME-TSO model syntax for your variables. To do this, you can use the file *"Mplus/ex_metso.inp"* as a template.

> **Tip**
>
> Make sure that you save the MCMC samples and plausible values of the factor scores with the **SAVEDATA** command and that you request the **TECH8** output.

3. Run the model directly in Mplus or from within R with the function `runModels()` from the package `MplusAutomation`. The advantage of running the model directly in Mplus is that it prints the progress. This is helpful to adjust your expectations on how long it will take to run the model. In contrast, this is not possible when running the model from within R.
4. Verify that your model converged by checking in the output file that the maximum *PSR* value is lower than 1.1 and relatively stable for the last 1000 iterations.
5. Use the custom function `reliability_metso()` to read the output into R and automatically compute all the reliability coefficients of interest. This function takes as input the name of the Mplus output file and a vector with the names of the items. Other arguments are explained in the source code.

> **Tip**
>
> Depending on the size of your data, the file where the MCMC of the plausible values are stored can be quite big, taking a couple of GB of storage. Because of this, the function `reliability_metso()` can be quite slow as it has to read such a big file into R.

The function `reliability_metso()` returns a list with two elements. The first one is a data.frame with the estimated reliability coefficient per person and item, and the second is an array that stores the MCMC samples of these coefficients.

References

Castro-Alvarez, S., Bringmann, L. F., Back, J., & Liu, S. (2025). The many reliabilities of psychological dynamics: An overview of statistical approaches to estimate the internal consistency reliability of intensive longitudinal data. *Psychological Methods*. https://doi.org/https://doi.org/10.1037/met0000778

Castro-Alvarez, S., Tendeiro, J., Jonge, P. de, Meijer, R. R., & Bringmann, L. (2022). Mixed-effects trait-state-occasion model: Studying the psychometric properties and the person–situation interactions of psychological dynamics. *Structural Equation Modeling: A Multidisciplinary Journal*, *29*(3), 438–451. https://doi.org/10.31234/osf.io/4ext3

Cattell, R. B. (1967). *The structuring of change by p-technique and incremental r-technique.* (C. W. Harris, Ed.; pp. 167–198). The University of Wisconsin Press.

Cranford, J. A., Shrout, P. E., Iida, M., Rafaeli, E., Yip, T., & Bolger, N. (2006). A Procedure for Evaluating Sensitivity to Within-Person Change: Can Mood Measures in Diary Studies Detect Change Reliably? *Personality and Social Psychology Bulletin*, *32*(7), 917–929. https://doi.org/10.1177/0146167206287721

Fuller-Tyszkiewicz, M., Hartley-Clark, L., Cummins, R. A., Tomyn, A. J., Weinberg, M. K., & Richardson, B. (2017). Using dynamic factor analysis to provide insights into data reliability in experience sampling studies. *Psychological Assessment*, *29*(9), 1120–1128. https://doi.org/10.1037/pas0000411

Geldhof, G. J., Preacher, K. J., & Zyphur, M. J. (2014). Reliability estimation in a multilevel confirmatory factor analysis framework. *Psychological Methods*, *19*(1), 72–91. https://doi.org/10.1037/a0032138

Hallquist, M. N., & Wiley, J. F. (2018). MplusAutomation: An R package for facilitating large-scale latent variable analyses in Mplus. *Structural Equation Modeling*, 621–638. https://doi.org/10.1080/10705511.2017.1402334

Hu, Y., Nesselroade, J. R., Erbacher, M. K., Boker, S. M., Burt, S. A., Keel, P. K., Neale, M. C., Sisk, C. L., & Klump, K. (2016). Test reliability at the individual level. *Structural Equation Modeling: A Multidisciplinary Journal*, *23*(4), 532–543. https://doi.org/10.1080/10705511.2016.1148605

Lai, M. H. C. (2021). Composite reliability of multilevel data: It's about observed scores and construct meanings. *Psychological Methods*, *26*(1), 90–102. https://doi.org/10.1037/met0000287

Leertouwer, Ij., Keijsers, L., Roekel, E. van, & Schuurman, N. K. (2025). *A practical guide to estimating reliability of intensive longitudinal data.* https://doi.org/10.31234/osf.io/uq4sk_v1

Molenaar, P. C. M. (1985). A dynamic factor model for the analysis of multivariate time series. *Psychometrika*, *50*(2), 181–202. https://doi.org/10.1007/BF02294246

Muthén, L. K., & Muthén, B. O. (1998-2025). *Mplus User's Guide. Eighth Edition.* Muthén & Muthén.

Nezlek, J. B. (2017). A practical guide to understanding reliability in studies of within-person variability. *Journal of Research in Personality*, *69*, 149–155. https://doi.org/10.1016/j.jrp.

2016.06.020

Ou, L., Hunter, M. D., & Chow, S.-M. (2019). What's for dynr: A package for linear and nonlinear dynamic modeling in R. *The R Journal*, *11*, 1–20.

R Core Team. (2025). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software*, *48*(2), 1–36. https://doi.org/10.18637/jss.v048.i02

Vogelsmeier, L. V. D. E., Uglanova, I., Rein, M. T., & Ulitzsch, E. (2024). Investigating dynamics in attentive and inattentive responding together with their contextual correlates using a novel mixture IRT model for intensive longitudinal data. *British Journal of Mathematical and Statistical Psychology*, *n/a*(n/a). https://doi.org/10.1111/bmsp.12373

William Revelle. (2025). *Psych: Procedures for psychological, psychometric, and personality research*. Northwestern University. https://CRAN.R-project.org/package=psych

Xiao, Y., Wang, P., & Liu, H. (2023). Assessing intra- and inter-individual reliabilities in intensive longitudinal studies: A two-level random dynamic model-based approach. *Psychological Methods*. https://doi.org/10.1037/met0000608

Yanagida, T. (2025). *Misty: Miscellaneous functions 't. yanagida'*. https://doi.org/10.32614/CRAN.package.misty