Appendix B.
Guide for the estimation of internal consistency in four scenarios using R.

This guide is recommended for beginners. See http://ddd.uab.cat/record/173917 for data bases and Table 1 and Table 2 for selected output. See additional details in main text.

To estimate internal consistency reliability in R, (a) prepare R for a working session (b) read the data to be analyzed and (c) perform the analysis in the three phases recommended in the main text. In Appendix B we describe the main features of R and the syntax lines you need to know in order to obtain the results in Table 1 and Table 2. The easiest way to run an example is to paste the syntax lines provided in Appendix A into R and if necessary, adapt them to your own analysis.

Working with R

If the free software environment R is not available on your computer, it can be downloaded free of charge at https://www.r-project.org/. The syntax provided in this Appendix can be used in any R interface, either the simple Rconsole or more developed interfaces such as RCommander, RStudio o DeduceR, which provide additional facilities besides the console.

To achieve results, launch R, wait for the prompt > to appear in the console, write a syntax line next to the prompt, press the enter key, and read the output below the syntax line. See below an extremely synthetic description of the R language, syntax features, file input/output, and installation commands. More information can be found at the R website (https://www.r-project.org/)

Regarding R language, you will use functions that read data and create objects containing the output. For example, when applied to quantitative data, the function reliability() produces an object containing α using Equation 3 and coefficient omega using Equation 5. All R functions are included in packages. For example, the package psych allows calculating Pearson correlation coefficients with the function lowerCor(), polychoric correlation coefficients with the function polychoric()and reliability coefficients with the function reliability(). Some packages are available as a default, but most must be installed and loaded before use. Finally, R has multiple packages and functions to carry out the same analysis (e.g., the CI for α can be obtained using the package psych or the package MBESS). We have selected some of them in the syntax provided in Appendix A.

Turning to the syntax features, R is case sensitive, so reliability(C1) is not the same as either reliability(c1) or as Reliability(C1). Among the special symbols to be found in the provided syntax, # denotes a comment that will not be evaluated by R but can be useful for human readers, <- is used to store a result into a new object, + - * / = are the obvious mathematical and logical operators and =~ is used to define factors in CFA. As for the naming conventions, R is somewhat flexible. Some names are camel case (e.g., semTools) others are separated by dots (e.g., install.packages) or use underscores (e.g., CFA_C1tau).

Regarding file input and output, it is useful to use a working directory defined by the user. Data files must be available at the working directory in order to be read using the syntax provided in Appendix A.

To prepare your working session, set the working directory and activate all necessary packages. The present working directory is found with the function:

```
getwd()
```

To change the working directory, the function setwd()should be used, indicating the new directory in brackets and quotation marks. Note that in R, directories are defined with the / slash instead of the usual \ bar. For example:

```
setwd("c:/workingdirectory")
```

The installation of the packages is done using the function install.packages()in which the name of the package is indicated in brackets and quotation marks.

In order to obtain the results in Table 1, we used the following packages: reshape2 to obtain the tables of frequencies or proportions and psych to obtain univariate statistics and Pearson or polychoric correlations. As for the results in Table 2, they were obtained using lavaan to specify, estimate and fit all measurement models, semTools to calculate the point estimates of coefficients omega and alpha and MBESS for the calculation of CI. So, the syntax reads:

```
install.packages("reshape2", dependencies = TRUE)
install.packages("psych", dependencies = TRUE)
install.packages("lavaan", dependencies = TRUE)
install.packages("semTools", dependencies = TRUE)
install.packages("MBESS", dependencies = TRUE)
```

When running this command, a list of repositories (CRAN mirror) can be displayed. Select one, preferably geographically close, and wait for the prompt > to appear in the console when the installation is finished. Once installed, the packages will remain in your local R files until removed using the function remove.packages()with the same conventions.

Every time a new working session is started, the packages must be loaded with the function library()indicating the name of the package in brackets:

```
library(reshape2)
library(psych)
library(lavaan)
library(semTools)
library (MBESS)
```

From that moment until the end of the working session, all functions of the loaded packages will be available. If you wish to obtain information about a particular package, for example, how to define their functions correctly, simply write the symbol ?? followed by the package name:

```
?? semTools
```

Reading data

Data can be read in different formats, but here we suggest using a simple text file. Table B1 shows a few lines of the data file of Case 1. Each line contains data from one respondent to all items and each column contains all responses to one of the six items. The values are separated using tab as a delimiter. The first row of the file contains a name for each item, in this case Y1, Y2, Y3, Y4, Y5 and Y6. This file was saved with the name of Case1.txt. During the analysis session, the data file must be available in the directory defined as working directory in R.

Table B1. First five records of case 1

Y1	Y2	Y3	Y4	Y5	Y6
2	2	3	3	2	1
3	4	2	3	3	4
4	4	3	4	4	3
3	2	4	3	3	3
1	3	2	3	3	2

This type of data file can be read with the function read_table (). In the bracket you should include two pieces of information, the name of the data file in quotation marks, and whether the first row contains (header=TRUE) or not (header=FALSE) the name of the variables. In our syntax the command was as follows:

```
C1<-read.table('Case1.txt', header=TRUE)
```

The content of the data file is transferred, by the symbol <-, to an object with a name specified by the user (in this example C1) for future reference. To check whether the table has been defined correctly simply write the name of the object and press the enter key:

```
C1
```

Conducting the analysis

Case 1: Analyzing essentially tau-equivalent measures

The R syntax necessary to conduct the three phases of the analysis and to achieve the results included in Table 1 (Phase 1) and Table 2 (Phase 2 and Phase 3) are described in turn.

Phase 1: Describing data

The following command would provide the table of response frequencies to each category for each item:

```
table(melt(C1))
```

We used the following command for the table of percentages:

```
prop.table(table(melt(C1)),1)*100
```

The basic descriptive statistics, such as mean, standard deviation, skewness and kurtosis were obtained by:

```
describeBy(C1)
```

The Pearson correlation matrix was obtained using:

```
lowerCor(C1, digits = 3)
```

From results in Table 1 we concluded that the responses to the items in Case 1 could be treated as quantitative, using ML estimation to test the measurement models. See the main text for a more detailed discussion.

Phase 2: Determining the best fitting measurement model

The essentially tau-equivalent measurement model was defined as follows:

```
C1tau <- 'Factor1 =~ L*Y1 + L*Y2 + L*Y3 + L*Y4 + L*Y5 + L*Y6'
```

where the latent variable (Factor1) is defined (= ~) as the weighted sum of the six items (Y_j). The weight (L) is a constant for all items to specify the assumption of essential tau-equivalence. The result of the analysis is transferred (<-) to an object that the user has named C1tau.

The command:

```
CFA_C1tau <- cfa(C1tau, C1,std.lv = TRUE)
```

performs a confirmatory factor analysis (cfa) under the model defined in C1tau on data stored in C1 . The results will be standardized (std.lv = TRUE) and stored (<-) in the object CFA_C1tau. The estimation method is ML by default.

The goodness of fit indices for the model were obtained as a summary of the object CFA_C1tau with the following command:

```
summary(CFA_C1tau, fit.measures=TRUE)
```

The congeneric measurement model was analogously defined and fitted, simply erasing the constant weights restriction and storing the result under a new user defined name (C1cong):

```
C1cong <- 'Factor1 =~ Y1 + Y2 + Y3 + Y4 + Y5 + Y6'
CFA_C1cong <- cfa(C1cong, C1,std.lv = TRUE)
summary(CFA_C1cong, fit.measures=TRUE)
```

As expected, the goodness of fit indices for both models (see Table 2) favored the tau-equivalent measurement model for Case 1, as discussed in detail in the main text.

Finally, it may be useful to keep in mind that if you wish to use a different estimator to the default, you must specify the desired estimator in quotation marks. For example, if you wish to use a robust estimator for quantitative data, the command would read:

```
CFA_C1cong <- cfa(C1cong, C1,std.lv = TRUE, estimator ="MLR")
```

Phase 3: Obtaining the reliability coefficients for essentially tau-equivalent measures

The output of the command:

```
reliability(CFA_C1tau)
```

provide various point estimates for reliability. Two were included as results for Case 1 in Table 2. The first is the Cronbach's alpha coefficient, labeled in the output as *alpha*, and calculated using Equation 4 with an ULS estimator. The third, labeled in the output as *omega2*, is obtained using a general formula for coefficient omega, the Equation 5. When applied to congeneric or tau-equivalent measures, such as those in Case 1, the result is equivalent to Equation 2 due to the fact that correlations between errors are all zero.

Alternatively, the commands:

```
ci.reliability(data=C1, type='alpha', interval.type='bsil', B=500)
ci.reliability(data=C1, type='alpha-CFA', interval.type='bsil', B=500)
```

provide interval estimates respectively for coefficients alpha and omega under the assumption of essentially tau equivalent measures. First, the data to be analyzed is specified with data=. Next, the internal consistency coefficient is specified with type=. The option 'alpha' stands for Cronbach's alpha coefficient obtained using Equation 4 with an ULS estimator. The option 'alpha-CFA' stands for the coefficient omega for tau-equivalent measures using Equation 2 with an ML estimator. The method used to estimate the standard error of measurement and therefore the CI is specified with interval.type=. The option in the example, 'bsil', uses bootstrap to calculate SE and logistic transformation is used to build CI. The number of bootstrap replications is defined in B=. The results can be seen in the column Phase 3 of Table 2. Alternatively, with large samples, the less computationally demanding delta method can be used, specifying interval.type='mll'for ML estimation with logistic transformation or interval.type='mlrl' for MLR estimation with logistic transformation.

As a conclusion, all reliability estimates were deemed to be within the accepted standards. See the main text for details.

Case 2: Analyzing congeneric measures

The results for Phase 1 and Phase 2 included in Table 1 and Table 2 were obtained using the data table for Case 2 (Case2.txt), and replacing C1 with C2 in the above syntax. As the best fitting model was the congeneric measurement model, the estimation of reliability coefficients during the Phase 3 changed with respect to Case 1. Only commands with changes are commented here.

The command:

```
reliability(CFA_C2cong)
```

provide both Cronbach's alpha and omega coefficients using the same estimation methods as in the previous case. Note that, even if tau-equivalence or at least high factor loadings are required for alpha to be correct, no warning is issued in the output. It is the researcher's responsibility to make decisions on the values to be published.

Interval estimations can be obtained applying the function ci.reliability() to the data C2, specifying type='alpha' for α and type='omega' for coefficient omega. The option type='omega' applies Equation 2 to the parameters estimated by ML under the congeneric measurement model and constitutes the main change with respect to the previous case. The complete commands read:

```
ci.reliability(data=C2, type='alpha', interval.type='bsil', B=500)

ci.reliability(data=C2, type='omega', interval.type='bsil', B=500)
```

We concluded that both alpha and omega were appropriate and very close estimates for reliability as expected due to the homogeneously high factor loadings of the congeneric measurement model. See the main text for details.

Case 3: Analyzing measures with correlated errors

The results for Phase 1 and Phase 2 included in Tables 1 and 2 were obtained using the appropriate data table named "Case3.txt" and replacing C1 with C3 in the above syntax. As neither tau-equivalent nor congeneric measurement models fitted the data, Phase 2 was completed testing a more flexible model which allowed some correlated error terms. The estimation of reliability coefficients during Phase 3 changed accordingly. Only the commands including changes are commented.

The correlation between errors was modelled as follows:

```
C3err_corr <- 'Factor1 =~ Y1 + Y2 + Y3 + Y4 + Y5 + Y6
Y4 ~~ Y5
```

```
Y4 ~~ Y6
Y5 ~~ Y6'
```

where ~~ is used to indicate the correlation between the error terms of items Y4, Y5 and Y6.

Again, the model was estimated and fitted with the usual commands:

```
CFA_C3err_corr <- cfa(C3err_corr, C3, std.lv = TRUE)
summary(CFA_C3err_corr, fit.measures=TRUE)
```

The point estimation of alpha and omega was obtained with:

```
reliability(CFA_C3err_corr)
```

As discussed in the main text, the value of alpha is an incorrect reliability estimate under the correlated errors model and was included in Table 2 only for illustration purposes. The correct estimator is the value labeled in the output as *omega2* obtained using Equation 5. Finally, the CI for omega was reported as not available in Table 2 due to the fact that presently no options for the ci.reliability()function allow calculation of the CI of omega coefficient in models with correlated errors.

Case 4: Analyzing ordinal data

The results for Phase 1 included in Table 1 were obtained using the appropriate data table named "Case4.txt" and replacing C1 with C4 in the above syntax. Although data came from responses to five point Likert scales, they were treated as ordinal due to the strong ceiling effects. Accordingly, the polychoric correlation matrix was obtained in Phase 1 using the command:

```
polychoric(C4)
```

In case items were dichotomous, the matrix of tetrachoric correlations could be obtained using the tetrachoric () function.

The specification of ordered categorical measurement models requires declaring ordinal variables using the option ordered=names() into the cfa() function. The complete analysis for categorical congeneric model would read:

```
C4cong <- 'Factor1 =~ Y1 + Y2 + Y3 + Y4 + Y5 + Y6'
CFA_C4cong <- cfa(C4cong, C4,std.lv = TRUE, ordered=names(C4))
summary(CFA_C4cong, fit.measures=TRUE)
```

In accordance with the ordinal nature of the data, the estimation of reliability coefficients during Phase 3 should change. The command:

```
reliability(CFA_C4cong)
```

calculate ordinal internal consistency coefficients so the value labeled as *alpha* in the output is the ordinal alpha defined in Equation 7. The value labeled as *omega3* is the nonlinear SEM reliability coefficient by Green and Yang. All other omega values in the output should be avoided as they are not interpretable values for categorical data. The correct estimate of reliability of Case 4 is the nonlinear SEM reliability, even if both alpha ordinal and nonlinear SEM reliability were included in Table 2 for illustration purposes.

Finally, the interval estimation of nonlinear based SEM reliability can be obtained with the function ci.reliability()and the options type='categorical' to define the categorical nature of the data and interval.type='bca' to select the bias corrected and accelerated bootstrap method. The whole syntax line would read:

```
ci.reliability(data=C4, type='categorical', interval.type='bca')
```