

Package ‘FCO’

February 13, 2026

Title Flexible Cutoffs for Model Fit Evaluation in Covariance-Based Structural Models

Version 2.0.1

Author Thomas Niemand [aut, cre] (ORCID:

<<https://orcid.org/0000-0001-7677-1168>>),

Robert Mai [ctb] (ORCID: <<https://orcid.org/0000-0002-8364-9328>>),

Nadine Schröder [ctb] (ORCID: <<https://orcid.org/0000-0001-6650-4656>>),

Andreas Falke [ctb] (ORCID: <<https://orcid.org/0000-0002-1519-0738>>)

Maintainer Thomas Niemand <thomas.niemand@gmail.com>

Description A toolbox to derive flexible cutoffs for fit indices in 'Covariance-based Structural Equation Modeling' based on the paper by 'Niemand & Mai (2018)' <[doi:10.1007/s11747-018-0602-9](https://doi.org/10.1007/s11747-018-0602-9)>. Flexible cutoffs are an alternative to fixed cutoffs - rules-of-thumb - regarding an appropriate cutoff for fit indices such as 'CFI' or 'SRMR'. It has been demonstrated that these flexible cutoffs perform better than fixed cutoffs in grey areas where misspecification is not easy to detect. The package provides an alternative to the tool at <<https://flexiblecutoffs.org>> as it allows to tailor flexible cutoffs to a given dataset and model, which is so far not available in the tool. The package simulates fit indices based on a given dataset and model and then estimates the flexible cutoffs. Some useful functions, e.g., to determine the 'GoF-' or 'BoF-nature' of a fit index, are provided. So far, additional options for a relative use (is a model better than another?) are provided in an exploratory manner.

License GPL (>= 3)

URL <https://flexiblecutoffs.org>, <https://github.com/ThomasNiemand/FCO/>

Depends R (>= 4.4.0)

Imports checkmate, cutpointr, data.table, doParallel, dplyr, foreach, ggplot2, lavaan, overlapping, parallel, PoisBinOrdNor, psych, rcompanion, simstandard, stringr, tidyr

Suggests knitr, MASS, rmarkdown, testthat

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8
LazyData true
RoxygenNote 7.3.3
NeedsCompilation no
Repository CRAN
Date/Publication 2026-02-13 20:30:02 UTC

Contents

bb1992	2
flex_co	3
flex_co2	5
gen_fit	7
gen_fit2	10
plot_fit2	15
pop_mod	16
recommend	18
recommend_dv	19
Index	21

bb1992	<i>Dataset from Babakus & Boller (1992)</i>
--------	---

Description

Data from Babakus & Boller (1992) who investigated the dimensionality of the SERVQUAL scale based on a sample of N = 502. The data is available as a data.frame (simulated via mvnrm in package MASS based on the correlation matrix provided by the authors) and used in the vignette.

Usage

data(bb1992)

Format

A data.frame of 22 variables (Q1-Q22) with 502 observations

Source

Provided in paper

References

Babakus, E., & Boller, G. W. (1992). An empirical assessment of the SERVQUAL scale. Journal of Business Research, 24(3), 253–268. [https://doi.org/10.1016/0148-2963\(92\)90022-4](https://doi.org/10.1016/0148-2963(92)90022-4)

Examples

```
data(bb1992)
head(bb1992, 3)
```

flex_co

*Obtain flexible cutoffs for one or two models***Description**

Obtain flexible cutoffs for one or two models

Usage

```
flex_co(fits, index, alpha.lev = 0.05, gof = NULL)
```

Arguments

fits	A list of simulated fit indices obtained from <code>gen_fit</code> . Based on the structure of fits, the number of models is derived.
index	A vector of fit indices or measures provided by function <code>fitmeasures</code> in package <code>lavaan</code>
alpha.lev	The predefined uncertainty For example, if the default uncertainty of .05 (5 percent) is accepted a-priori, the 5 percent <code>stats::quantile</code> (of type 8, see <code>?stats::quantile</code>) of the simulated distribution for correctly specified CFA models with the given model and sample characteristics determines the flexible cutoff. Options are .001, .01, .05, and .10. Higher values are more conservative.
gof	An optional vector as to whether the indices are GoF (Goodness-of-Fit index)? If TRUE, a GoF is assumed. If FALSE, a BoF is assumed. Depending on the nature of the underlying fit index, the appropriate lower (GoF) or upper (BoF) width of the respective confidence interval as defined by the <code>stats::quantile</code> is used to derive the flexible cutoff. If not provided or not equal to the number of fit indices, the function guesses the type for known fit indices (e.g., SRMR is a BoF).

Value

A list of information regarding the selected fit index providing its flexible cutoff for the given parameters.

Examples

```
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
#A single model to obtain fit indices for
mod <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
```

```

F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
"

fits.single <- gen_fit(mod1 = mod, x = bb1992, rep = 10, standardized = FALSE)
flex_co(fits = fits.single, index = c("CFI", "SRMR"))

#Two models, an unconstrained and a constrained model to compare fit indices
mod.con <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
F1 ~~ 0 * F2
"

fits.con <- gen_fit(
  mod1 = mod,
  mod2 = mod.con,
  x = bb1992,
  rep = 10
)
flex_co(fits = fits.con,
  index = c("CFI", "SRMR"),
  alpha.lev = .05)

#Two models for discriminant validity testing, this resembles constraining with a cutoff of .9
fits.dv.con <- gen_fit(
  mod1 = mod,
  x = bb1992,
  rep = 10,
  dv = TRUE,
  dv.factors = c("F4", "F5"),
  dv.cutoff = .9
)
flex_co(fits = fits.dv.con,
  index = "CFI",
  alpha.lev = .05)
mod.dv.con <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
F4 ~~ .9 * F5
"

lavaan::fitmeasures(
  lavaan::cfa(
    model = mod.dv.con,
    data = bb1992,
    auto.fix.first = FALSE,
    std.lv = TRUE
  ),
  fit.measures = "cfi"

```

```

)
#Two models for discriminant validity testing, this resembles merging.
fits.dv.merge <- gen_fit(
  mod1 = mod,
  x = bb1992,
  rep = 10,
  dv = TRUE,
  dv.factors = c("F4", "F5"),
  merge.mod = TRUE)

flex_co(fits = fits.dv.merge,
index = "CFI",
alpha.lev = .05)
mod.dv.merge <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17 + Q6 + Q14 + Q15 + Q16
"
lavaan::fitmeasures(
  lavaan::cfa(
    model = mod.dv.merge,
    data = bb1992
  ),
  fit.measures = "cfi"
)

```

flex_co2

*Obtain cutoffs for fit indices simulated by gen_fit2.***Description**

Obtain cutoffs for fit indices simulated by gen_fit2.

Usage

```

flex_co2(
  fits = NULL,
  correct.fits = NULL,
  miss.fits = NULL,
  index = "CFI",
  alpha = c(0.05),
  beta = c(0.05, 0.1)
)

```

Arguments

fits	The object returned from gen_fit2 which is a list of correct fit values and mis-specified fit values.
------	---

correct.fits	For compatibility reasons, the correct fit values can be defined separately.
miss.fits	For compatibility reasons, the misspecified fit values can be defined separately.
index	A vector of length ≥ 1 with names of fit indices the user wishes to explore. Capitalization does not matter, either e.g., CFI or cfi are accepted. Default is CFI as it might be easier to understand.
alpha	The acceptable Type I error representing the empirical quantile p, see details in gen_fit2. Multiple values can be provided as a vector. Default is .05.
beta	The acceptable Type II error representing the empirical quantile p, see details in gen_fit2. Multiple values can be provided as a vector. Default is c(.05, .10).

Details

This function returns cutoffs for the the fit indices simulated by gen_fit2. For details, please refer to gen_fit2. Please note that the results are only based on the simulation of the model specified and its misspecified variant under the conditions of the gen_fit2 function.

Value

A list consisting of a tibble for the empirical quantiles estimated for the provided alpha, beta values and indices, a tibble for the derived cutoff values for each parameter (alpha, beta, approach, index, cutoff), a tibble for the evaluation (also True Negatives, False Positives, True Positives, False Negatives, Type I error, Type II error, Sum of both Types, Power and Specificity), a vector for the notation on the evaluation tibble, and a tibble displaying overlap statistics for each index (Overlap percentage, AUC, U-test).

Examples

```
#Simple example
library(lavaan)
library(dplyr)
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(
  HS.model,
  data = HolzingerSwineford1939
)
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
fits <- gen_fit2(fit = fit, rep = 100)
#Default evaluation:
flex_co2(fits)
#Changed alpha and beta values:
flex_co2(fits, alpha = .05, beta = .05)
flex_co2(fits, alpha = .10, beta = .20)
#Different fit indices:
flex_co2(fits, index = c("CFI", "SRMR", "RMSEA"))
```

gen_fit

Obtain fit statistics from one or two models

Description

Obtain fit statistics from one or two models

Usage

```
gen_fit(  
  mod1 = NULL,  
  mod2 = NULL,  
  x = NULL,  
  n = NULL,  
  rep = 500,  
  type = "NM",  
  dv = FALSE,  
  dv.factors = NULL,  
  merge.mod = FALSE,  
  dv.cutoff = 0.9,  
  standardized = TRUE,  
  assume.mvn = TRUE,  
  multi.core = TRUE,  
  cores = 2,  
  seed = 1111,  
  pop.mod1 = NULL,  
  pop.mod2 = NULL  
)
```

Arguments

mod1	A lavaan model to specify the CFA.
mod2	Another lavaan model for a model comparison. If missing and merge.mod = TRUE, a merged model from function merge_factors is estimated based on mod1.
x	A dataset for the model of nrow observations (minimum: 50) and ncol indicators (minimum: 4)
n	A sample size specified instead of a dataset (minimum: 50, maximum: 50000). Requires a population model via pop.mod1.
rep	Number of replications to be simulated (default: 500, minimum: 10, maximum: 5000)
type	Type of underlying population model. Based on the model(s) provided, a population model is derived to simulate the fit indices by function pop_mod. The type determines the factor loadings and covariances assumed for this population model. NM (the default when only one model is provided): Uses the factor

	loadings and covariances from Niemand & Mai's (2018) simulation study. HB: Uses the factor loadings and covariances from Hu & Bentler's (1999) simulation study. EM: Empirical (the default when two models are provided or merge.mod is TRUE), uses the given factor loadings and covariances.
dv	Should the fit statistics be calculated for discriminant validity testing? If no (the default), this is not assumed. If yes, consider the arguments of merge.mod, dv.factors and cutoff. So far, two options of discriminant validity testing are supported. Constraining: A factor correlation between two factors can be constrained as selected by the dv.factors argument. In this case, dv.cutoff applies and merge.mod is not required. Merging: Two factors can be merged into one, again controlled by the dv.factors argument. In this case, merge.mod applies and dv.cutoff is not required (as cutoff = 1 is implied).
dv.factors	Names of the factors to be considered. Must be equal to 2. If missing (the default), the first and second factor of the model are selected.
merge.mod	This is used for merging. If FALSE (the default), fit measures for mod1 are estimated for a single model as long as no mod2 is provided. If TRUE, a merged model from function merge_factors is estimated based on mod1. In this case, no mod2 is required.
dv.cutoff	This is used for constraining. It determines the critical correlation assumed to be a cutoff for discriminant validity testing. For example, based on Rönkkö & Cho (2020), a cutoff of .9 indicates a severe issue in discriminant validity between the selected factors. Cutoffs between .8 and 1 are recommended. The function returns a warning, if the cutoff is below .8.
standardized	Are factor loadings assumed to be standardized and covariances to be correlations (default: TRUE)?
assume.mvn	Should multivariate normality (mvn) be assumed? If TRUE (the default), kurtosis and skewness are set to 1 for simulated data. If FALSE, kurtosis and skewness are estimated from dataset x via semTools::mardiaKurtosis and semTools::mardiaSkew.
multi.core	Should multiple cores be used to simulate fit indices? If TRUE (the default), mclapply (on Linux or Mac machines) or parLapply (on Windows machines) from parallel package with the number of specified cores is used. If FALSE, a single core is used.
cores	How many cores should be used for multiple cores? The default is 2. Consider the available number of cores of your system.
seed	The seed to be set to obtain reproducible cutoffs (default: 1111). Defines a vector of length rep with the seed being the first value.
pop.mod1	For flexibility reasons, an optional lavaan population model can be provided. This is required together with n if x is missing.
pop.mod2	Another optional lavaan population model.

Value

A list of simulated fit statistics (fco) and all previously defined parameters.

References

- Hu, L., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling*, 6(1), 1–55. <https://doi.org/10.1080/10705519909540118>
- Niemand, T., & Mai, R. (2018). Flexible cutoff values for fit indices in the evaluation of structural equation models. *Journal of the Academy of Marketing Science*, 46(6), 1148—1172. <https://doi.org/10.1007/s11747-018-0602-9>
- Rönkkö, M., & Cho, E. (2020). An updated guideline for assessing discriminant validity. *Organizational Research Methods*. <https://doi.org/10.1177/1094428120968614>

Examples

```
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
#A single model to obtain fit indices for
mod <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
"

fits.single <- gen_fit(mod1 = mod, x = bb1992, rep = 10, standardized = FALSE)

#Two models, an unconstrained and a constrained model to compare fit indices
mod.con <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
F1 ~~ 0 * F2
"

fits.con <- gen_fit(
  mod1 = mod,
  mod2 = mod.con,
  x = bb1992,
  rep = 10
)
#Two models for discriminant validity testing, this resembles constraining with a cutoff of .9
fits.dv.con <- gen_fit(
  mod1 = mod,
  x = bb1992,
  rep = 10,
  dv = TRUE,
  dv.factors = c("F4", "F5"),
  dv.cutoff = .9
)

#Two models for discriminant validity testing, this resembles merging.
fits.dv.merge <- gen_fit(
```

```

mod1 = mod,
x = bb1992,
rep = 10,
dv = TRUE,
dv.factors = c("F4", "F5"),
merge.mod = TRUE
)

```

gen_fit2

Obtain fit statistics from correctly and misspecified models.

Description

Obtain fit statistics from correctly and misspecified models.

Usage

```

gen_fit2(
  fit = NULL,
  mod = NULL,
  x = NULL,
  model.type = "111",
  pop.mod = NULL,
  n = NULL,
  rep = 500L,
  type = "NM",
  cfa = TRUE,
  data.types = NULL,
  esti = "ML",
  cores = 2,
  standardized = TRUE,
  es.lam = "low",
  es.cor = "large",
  es.f2 = "moderate",
  sk = 0,
  ku = 1,
  seed = 1111,
  random = TRUE
)

```

Arguments

fit	A fitted object from lavaan.
mod	A lavaan model to specify the CFA. If a fitted lavaan object is provided, the model is taken from this object via lavaan::parTable.
x	A dataset for the model of nrow observations and ncol indicators. If a fitted lavaan object is provided, the dataset is taken from this object.

model.type	A model type defining the number of structural model misspecifications (first integer), measurement model misspecifications (second integer) and residual covariance misspecifications (third integer) assumed for the misspecified model. Can be written as a character (default: "111") or as an integer vector (default: c(1,1,1)). For each integer, a maximum of 3 is supported so far. For example, "100" refers to one structural model misspecification (factor correlations set to 0), zero measurement model misspecifications (cross-loadings set to 0) and zero residual covariances introduced to the correct model described in mod. A "000" model is not allowed as it would be identical to the correct model.
pop.mod	For flexibility reasons, an optional lavaan population model can be provided. This is required together with n if x is missing or no fitted lavaan object is used.
n	A sample size specified instead of a dataset. Requires a population model via pop.mod. If a fitted lavaan object is provided, the sample size is taken from this object.
rep	Number of replications to be simulated (default: 500).
type	Type of underlying population model. Based on the model provided, a population model is derived to simulate the fit indices by function the internal function pop_mod. The type determines the factor loadings and covariances assumed for this population model. NM (the default): Uses the factor loadings and covariances from Niemand & Mai's (2018) simulation study. HB: Uses the factor loadings and covariances from Hu & Bentler's (1999) simulation study. EM: Empirical uses the given factor loadings and covariances. The underlying population model is used to derive a misspecified population model based on the model provided in model.type.
cfa	If TRUE (the default), the population model is generated assuming a CFA, i.e., there are only loadings and correlations. If FALSE, the model can be a regression model or any other type of SEM. In this case, the argument type determines how the population model is built. In case of "EM", the population model is defined from the model with the given parameters. If type is "NM" or "HB", the population model is defined based on the loadings, correlations and regression coefficients (beta) given effect size values for these three (loadings: see es.lam; correlations: see es.cor; beta: see es.f2). In the latter cases, aco and afl are not used.
data.types	Types of the manifest variables. Users can specify a vector of the length of variables with C = count, B = binary, O = ordinal, N = normal in the same order as in the dataset. These types are then used to simulate data for the population model based on median (count), mean (binary), cumulative relative frequencies of values (ordinal) as well as mean and variance (normal) applying the PoisBinOrdNor::intermat and PoisBinOrdNor::genPBONdata functions from package PoisBinOrdNor. That is, categorical and binary variables are also supported. Argument type is set to EM when data.types are defined (otherwise, normal data would be implied).
esti	The estimator to be used for model estimation in lavaan, defaults to "ML". Consider changing when needed. If a fitted lavaan object is provided, the estimator is taken from this object.
cores	How many cores should be used for multiple cores? The default is 2. Consider the available number of cores of your system.

standardized	Are factor loadings assumed to be standardized and covariances to be correlations (default: NULL)? The internal function pop_mod checks this feature and returns a warning if set to TRUE (any > 1) or FALSE (all < 1). Otherwise, TRUE or FALSE is guessed from the loadings (all < 1 leads to TRUE, any > 1 to FALSE).
es.lam	Effect size assumed for the loadings when cfa = FALSE and type is not “EM”. Options are ‘low’ (.7), ‘moderate’ (.8) and ‘large’ (.9). Defaults to ‘low’. The loadings are equal for type “NM” and vary according to Hu & Bentler (1999) for “HB”.
es.cor	Effect size assumed for the correlations when cfa = FALSE and type is not “EM”. Options are ‘low’ (.1), ‘moderate’ (.3) and ‘large’ (.5) based on Cohen’s (1988) conventions. Defaults to ‘large’. The correlations are equal for type “NM” and vary according to Hu & Bentler (1999) for “HB”.
es.f2	Effect size assumed for the regression coefficients when cfa = FALSE and type is not “EM”. Options are ‘low’ (.02), ‘moderate’ (.15) and ‘large’ (.35) based on Cohen’s (1988) conventions. Defaults to ‘large’. The regression coefficients are equal for type “NM” and “HB”.
sk	Should skewness (default: 0) be assumed to indicate multivariate normality? In case of excessive skewness via psych::skew, a warning is provided and the user can enter a more appropriate sk for the data in the next run.
ku	Should kurtosis (default: 1) be assumed to indicate multivariate normality? In case of excessive kurtosis via psych::kurtosi, a warning is provided and the user can enter a more appropriate ku for the data in the next run.
seed	The seed to be set to obtain reproducible cutoffs (default: 1111). Defines a vector of length rep with the seed being the first value.
random	Should the misspecified population model be generated randomly (default: TRUE)? To avoid a bias by always misspecifying the same parameter in the model based on model.type for every replication (= FALSE), the parameter can be randomly selected (= TRUE). Results differ slightly, yet random = TRUE is a bit slower.

Details

Originally proposed by Niemand & Mai (2018), flexible cutoffs (hereafter FCO1) have been developed with only a correct model in mind. This simple, first approach of simulated cutoffs for fit indices hence cannot consider misspecified models under consideration. To improve on this, multiple decision rules are introduced that – based on a misspecified model – allow to determine the Type II error of misfit for a given cutoff and fit indicator. With this feature in mind, this enables different decision rules (i.e., what type of error is considered and how) based on pertinent literature: FCO1: The original proposed flexible cutoffs (Niemand & Mai, 2018), only considering Type I error. The p (e.g., 5%) quantile of all simulated correct models is taken (assuming a GoF like CFI, alternatively 1-p for a BoF like SRMR). FCO2: A modified flexible cutoff considering Type I and II error. The 1-p (e.g., 95%) quantile of all simulated misfit models is taken when this quantile is smaller than or equal to the p (e.g., 5%) quantile. Otherwise, the p (e.g., 5%) quantile of all simulated correct models is taken. FCO2 (alike FCO1) always provides a cutoff (assuming a GoF like CFI, alternatively p (misfit model) and 1-p (correct model) for a BoF like SRMR) DFI: A modified dynamic cutoff considering Type I and II error (McNeish & Wolff, 2023). The 1-p (e.g., 95%) quantile of all simulated

misfit models is taken when this quantile is smaller than or equal to the p (e.g., 5%) quantile. Otherwise, no cutoff is provided (NA). The DFI-decision rule tends to provide no cutoff when correct and misfit models overlap strongly (assuming a GoF like CFI, alternatively p (misfit model) and $1-p$ (correct model) for a BoF like SRMR). CP: A modified cutoff considering Type I and II error via an optimal cutpoint (Groskurth et al., 2025). The cutoff is found by taking the cutoff with the highest sum of sensitivity ($1 - \text{Type I error}$) and $1 - \text{specificity}$ (Type II error) derived from simulated correct and misfit models via `cutpointr::cutpointr`. CP always provides a cutoff. Fix: Fixed cutoffs (Hu & Bentler, 1999) are also provided for comparison. Since it cannot be objectively determined what level or type of misspecification (and to which extent) demarcates "acceptable" and "unacceptable" misfit, generalizability of the misspecification procedure becomes a vital question. To overcome this issue, a PROCESS-like (Hayes, 2017) approach is proposed. Instead of expecting that the user provides an appropriate misspecified model (as in `simsem`), which might be highly user-unfriendly, the user only provides a type of misfit model via `model.type`. This argument defines the number of structural model misspecifications (first integer), measurement model misspecifications (second integer) and residual covariance misspecifications (third integer) assumed for the misfit model. For example, "100" refers to one structural model misspecification (factor correlations set to 0), zero measurement model misspecifications (no cross-loadings set to 0) and zero residual covariances introduced to the correct model described in `mod`. The default is set to "111", which corresponds to a model where one correlation, one cross-loading and one residual covariance may be overlooked. If a researcher is certain, that only one type of misspecification is important, the value can be changed to a "100", "010", or "001" model for example. Comparing multiple `model.type` specifications is recommended. Based on feedback on the previous versions of FCO, some new features are integrated, most importantly direct input of a fitted lavaan object (`fit`), a one-step calculation of fits, support for different types of variables, extensive checking of the model and data characteristics by an internal function providing better warning and error descriptions, directly setting skewness (`sk`) and kurtosis (`ku`), easier parallelization options and random generation (`random`) of the misfit model. Further, two more functions have been introduced to better compare the implications (e.g., in terms of implied Type I and II errors) across decision rules and to visualize the simulated correct and misfit model distributions. Function `flex_co2` now provides detailed information on the quantiles, cutoffs (all decision rules plus fixed cutoffs), evaluation (including sum of errors, Type I error, Type II error estimates from the simulation data), a notation and overlap statistics (percentage overlap from `overlapping::overlap`, AUC from `cutpointr::cutpointr` and a U-test converted to d from `rcompanion::wilcoxonR`) to identify the degree of overlap between simulated correct and misfit model distributions. Thereby, users can select an appropriate cutoff, inspect which decision rule works best for their data and model and determine how much the distributions overlap. Function `plot_fit2` complements these features and plots the simulated distributions for correct and misfit models to illustrate the distributions, their overlap and the cutoff quantiles. Multiple notes are given regarding the estimates 1) "DFI" does not refer to using the same approach as `dynamic::cfaHB` as proposed by McNeish & Wolf (2023). DFI only refers to the authors' decision rule, but applying the same PROCESS-like approach as FCO1, FCO2 and CP. 2). For compatibility reasons, functions `gen_fit`, `flex_co`, `pop_mod`, `recommend` and `recommend_dv` are (virtually) unchanged and only apply for FCO1. Likewise, functions `flex_co2` and `plot_fit2` only apply for results of this function `gen_fit2`.

Value

A list of simulated fit statistics for correct models (`correct.fits`) and misfit models (`miss.fits`).

References

- Groskurth, K., Bhaktha, N., & Lechner, C. M. (2025). The simulation-cum-ROC approach: A new approach to generate tailored cutoffs for fit indices through simulation and ROC analysis. *Behavior Research Methods*, 57(5), Article 135. doi:10.3758/s1342802502638x
- Hu, L., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural Equation Modeling*, 6(1), 1–55. doi:10.1080/10705519909540118
- McNeish, D., & Wolf, M. G. (2023). Dynamic fit index cutoffs for confirmatory factor analysis models. *Psychological Methods*, 28(1), 61–88. doi:10.1037/met0000425
- Niemand, T., & Mai, R. (2018). Flexible cutoff values for fit indices in the evaluation of structural equation models. *Journal of the Academy of Marketing Science*, 46(6), 1148–1172. doi:10.1007/s1174701806029

Examples

```
#Simple example
library(lavaan)
library(dplyr)
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(
  HS.model,
  data = HolzingerSwineford1939
)
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
fits <- gen_fit2(fit = fit, rep = 100)
flex_co2(fits)
plot_fit2(fits)

#Different data types
dat <- lavaan::HolzingerSwineford1939
cdat <- dplyr::select(dat, x1, x2, x3, x4, x5, x6, x7, x8, x9)
#For demo purposes, some variables are changed:
cdat <- cdat %>% mutate(
  x1 = round(x1, digits = 0),
  x3 = round(x3, digits = 0),
  x2 = ifelse(x2 > 4, 1, 0),
  x4 = ifelse(x4 > 2, 1, 0),
  x5 = round(x5, digits = 0),
  x8 = round(x8, digits = 0)
)
cfits <- cfa(model = HS.model, data = cdat)
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
cfits <- gen_fit2(fit = cfits,
  data.types = c("C", "B", "C", "B", "O", "N", "N", "O", "N"), rep = 100)
flex_co2(cfits)
plot_fit2(cfits)
```

```
#Multiple fit indices
flex_co2(fits, index = c("cfi", "SRMR", "RMSEA"))
plot_fit2(fits, index = c("cfi", "SRMR", "RMSEA"))
```

plot_fit2

Plotting the distributions of selected simulated fit indices

Description

Plotting the distributions of selected simulated fit indices

Usage

```
plot_fit2(
  fits = NULL,
  correct.fits = NULL,
  miss.fits = NULL,
  index = "CFI",
  alpha = 0.05,
  beta = 0.1
)
```

Arguments

<code>fits</code>	The object returned from <code>gen_fit2</code> which is a list of correct fit values and misspecified fit values.
<code>correct.fits</code>	For compatibility reasons, the correct fit values can be defined separately.
<code>miss.fits</code>	For compatibility reasons, the misspecified fit values can be defined separately.
<code>index</code>	A vector of length ≥ 1 with names of fit indices the user wishes to explore. Capitalization does not matter, either e.g., CFI or cfi are accepted. Default is CFI as it might be easier to understand.
<code>alpha</code>	The acceptable Type I error representing the empirical quantile p, see details in <code>gen_fit2</code> . Multiple values can be provided as a vector. Default is .05.
<code>beta</code>	The acceptable Type II error representing the empirical quantile p, see details in <code>gen_fit2</code> . Multiple values can be provided as a vector. Default is <code>c(.05, .10)</code> .

Details

For details, please refer to `gen_fit2`. Please note that the results are only based on the simulation of the model specified and its misspecified variant under the conditions of the `gen_fit2` function.

Value

A `ggplot2` object with the simulated cutoffs for correct and misspecified models iterated across the number of indices provided.

Examples

```
#Simple example
library(lavaan)
library(dplyr)
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- cfa(
  HS.model,
  data = HolzingerSwineford1939
)
#Note: Demonstration only! Please use higher numbers of replications (>= 500).
fits <- gen_fit2(fit = fit, rep = 100)
#Default plot:
plot_fit2(fits)
#Changed alpha and beta values:
plot_fit2(fits, alpha = .05, beta = .05)
plot_fit2(fits, alpha = .10, beta = .20)
#Different fit indices:
plot_fit2(fits, index = c("CFI", "SRMR", "RMSEA"))
```

pop_mod

Helper function to obtain population model for simulation based on data and model

Description

Helper function to obtain population model for simulation based on data and model

Usage

```
pop_mod(
  mod,
  x,
  type = "NM",
  data.types = NULL,
  standardized = NULL,
  afl = 0.7,
  aco = 0.3,
  seed = seed
)
```

Arguments

mod	A lavaan model (only CFA supported so far)
x	A dataset for the model of nrow observations (minimum: 50) and ncol indicators (minimum: 4)

type	Type of population model. NM (the default): Uses the factor loadings and covariances from Niemand & Mai's (2018) simulation study. HB: Uses the factor loadings and covariances from Hu & Bentler's (1999) simulation study. EM: Empirical, uses the given factor loadings and covariances. EM is not recommended for confirmative use as it leads to the least generalizable cutoffs.
data.types	Types of the manifest variables. Users can specify a vector of the length of variables with C = count, B = binary, O = ordinal, N = normal in the same order as in the dataset. These types are then used to simulate data for the population model based on median (count), mean (binary), cumulative relative frequencies of values (ordinal) as well as mean and variance (normal) applying the <code>PoisBinOrdNor::intermat</code> and <code>PoisBinOrdNor::genPBONdata</code> functions from package <code>PoisBinOrdNor</code> . That is, categorical and binary variables are also supported. Argument type is set to EM when data.types are defined (otherwise, normal data would be implied).
standardized	Are factor loadings assumed to be standardized and covariances to be correlations (default: TRUE)?
af1	Average factor loading of indicators per factor, only relevant for type = "NM" (default: .7).
aco	Average correlation between factors, only relevant for type = "NM" (default: .3).
seed	The seed to be set to obtain reproducible cutoffs (default: 1111). Defines a vector of length rep with the seed being the first value.

Value

List of population model type, standardized, average factor loading and average correlation. All values are round to three decimals.

Examples

```
mod <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
"
pop_mod(mod, x = bb1992, type = "NM")$pop.mod
pop_mod(mod, x = bb1992, type = "HB")$pop.mod
pop_mod(mod, x = bb1992, type = "EM")$pop.mod
pop_mod(mod, x = bb1992, type = "NM", af1 = .9)$pop.mod
pop_mod(mod, x = bb1992, type = "NM", aco = .5)$pop.mod
pop_mod(mod, x = bb1992, type = "EM", standardized = FALSE)$pop.mod
```

recommend

Obtain recommendations based on Mai et al. (2021)

Description

This function recommends pre-defined selected fit indices in case the user does not know which fit index should be used for model evaluation. Results may differ based on three settings, the sample size of the data, the research purpose of the investigated model and the focus of the model. For obvious reasons, this function only works for single models and does not accept any other model type.

Usage

```
recommend(
  fits,
  purpose = "novel",
  focus = "cfa",
  override = FALSE,
  index = NULL,
  digits = 3
)
```

Arguments

<code>fits</code>	A list of simulated fit indices obtained from <code>gen_fit</code> . Based on the structure of <code>fits</code> , the number of models is derived.
<code>purpose</code>	The research purpose of the model investigated. Is the underlying model novel (default) or established (= established). This parameter is relevant to find the proper recommended fit indices.
<code>focus</code>	The focus of estimation for the model. Is the focus on CFA (default) or analyzing the structural model of a theoretical model (= structural)? This parameter is relevant to find the proper recommended fit indices.
<code>override</code>	Should the recommendations by Mai et al. (2021) overridden (default: FALSE)? This may be useful to explore models outside of the scope of the paper. In this case, the recommended fit indices are not determined by the function, and hence need to be provided. In this case, the function requires the argument <code>index</code> .
<code>index</code>	An optional vector of fit indices or measures provided by function <code>fitmeasures</code> in package <code>lavaan</code> . This argument is required when <code>override</code> is TRUE. It is ignored otherwise.
<code>digits</code>	An optional integer to round fit values and cutoffs (min: 1, max: 5).

Value

A list of information regarding the recommended fit indices based on Mai et al. (2021) or when overridden, based on the provided indices.

References

Mai, R., Niemand, T., & Kraus, S. (2021). A Tailor-Fit Model Evaluation Strategy for Better Decisions about Structural Equation Models, *Technological Forecasting & Social Change*, 173(December) 121142. <https://doi.org/10.1016/j.techfore.2021.121142>

Examples

```
#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
mod <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
"

fits.single <- gen_fit(mod1 = mod, x = bb1992, rep = 10, standardized = FALSE)
recommend(fits.single)
recommend(fits.single, purpose = "established")
recommend(fits.single,
          override = TRUE,
          index = c("CFI", "SRMR"))
```

recommend_dv

Obtain recommendations for discriminant validity testing

Description

This function recommends on potential issues for discriminant validity testing, based on differences between fit values and differences between flexible cutoffs. Two approaches of testing are supported: merging and constraining.

Usage

```
recommend_dv(fits, index = "CFI", digits = 3)
```

Arguments

fits	A list of simulated fit indices obtained from <code>gen_fit</code> . Based on the structure of fits, the number of models is derived.
index	A vector of fit indices or measures provided by function <code>fitmeasures</code> in package <code>lavaan</code> . The default is set to CFI.
digits	An optional integer to round fit values and cutoffs (min: 1, max: 5).

Value

A list of information regarding discriminant validity testing.

Examples

```

#Note: Demonstration only! Please use higher numbers of replications for your applications (>= 500).
mod <- "
F1 =~ Q5 + Q7 + Q8
F2 =~ Q2 + Q4
F3 =~ Q10 + Q11 + Q12 + Q13 + Q18 + Q19 + Q20 + Q21 + Q22
F4 =~ Q1 + Q17
F5 =~ Q6 + Q14 + Q15 + Q16
"

#Two models for discriminant validity testing, this resembles constraining with a cutoff of .9
fits.dv.con <- gen_fit(
  mod1 = mod,
  x = bb1992,
  rep = 10,
  dv = TRUE,
  dv.factors = c("F4", "F5"),
  dv.cutoff = .9
)
recommend_dv(fits.dv.con)

#Two models for discriminant validity testing, this resembles merging.
fits.dv.merge <- gen_fit(
  mod1 = mod,
  x = bb1992,
  rep = 10,
  dv = TRUE,
  dv.factors = c("F4", "F5"),
  merge.mod = TRUE
)
recommend_dv(fits.dv.merge)

```

Index

* **datasets**

bb1992, [2](#)

bb1992, [2](#)

flex_co, [3](#)

flex_co2, [5](#)

gen_fit, [7](#)

gen_fit2, [10](#)

plot_fit2, [15](#)

pop_mod, [16](#)

recommend, [18](#)

recommend_dv, [19](#)