Package 'powRICLPM'

July 23, 2025

Title Perform Power Analysis for the RI-CLPM and STARTS Model

Version 0.2.1

Date 2024-10-26

Maintainer Jeroen Mulder <j.d.mulder@uu.nl>

Description Perform user-friendly power analyses for the random intercept cross-lagged panel model (RI-CLPM) and the bivariate stable trait autoregressive trait state (STARTS) model. The strategy as proposed by Mulder (2023) <doi:10.1080/10705511.2022.2122467> is implemented. Extensions include the use of parameter constraints over time, bounded estimation, generation of data with skewness and kurtosis, and the option to setup the power analysis for Mplus.

License MIT + file LICENSE

URL https://jeroendmulder.github.io/powRICLPM/

BugReports https://github.com/JeroenDMulder/powRICLPM/issues/

Depends R (>= 4.0.0), stats, utils

Imports cli, future.apply, lavaan (>= 0.6.7), lifecycle, progressr, rlang, ggplot2, future

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Jeroen Mulder [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-5553-0856>), Netherlands Organization for Scientific Research [fnd]

Repository CRAN

Date/Publication 2024-10-26 13:50:03 UTC

Contents

`check_Phi` *Check Interpretation* `Phi` *Argument*

Description

Write a textual interpretation of the values in `Phi`. This can be used to check if `Phi` has been correctly specified.

Usage

```
check_Phi(Phi)
```

Arguments

`Phi`	A matrix, with standardized autoregressive effects (on the diagonal) and cross-lagged effects (off-diagonal) in the population. Columns represent predictors and rows represent outcomes.

Value

No return value, called for side effects.

Examples

```
# Correctly specified `Phi`
Phi1 <- matrix(c(.4, .1, .2, .3), ncol = 2, byrow = TRUE)
check_Phi(Phi1)

# `Phi` with too large standardized effects
Phi2 <- matrix(c(.6, .5, .4, .7), ncol = 2, byrow = TRUE)
Phi2 <- check_Phi(Phi2)
```

give *Extract Information From* powRICLPM *Object*

Description

Extract information stored within a powRICLPM object (internally used by print.powRICLPM and summary.powRICLPM). See "Details" for which pieces of information can be extracted. The information is presented by condition (i.e., sample size, number of time points, and ICC).

Usage

```
give(from, what, parameter = NULL)
```

Arguments

from	A powRICLPM object
what	A character string, denoting the information to extract, either "conditions", "estimation_problems", "results", or "names".
parameter	(optional) When what = "results", a character string denoting the parameter to extract the results for.

Details

The following information can be extracted from the powRICLPM object:

- conditions: A data.frame with the different experimental conditions per row, where each condition is defined by a unique combination of sample size, number of time points and ICC.
- estimation_problems: The proportion of fatal errors, inadmissible values, or non-converged estimations (columns) per experimental conditions (row).
- results: The average estimate (average), minimum estimate (minimum), standard deviation of parameter estimates (SD), the average standard error (SEavg), the mean square error (MSE), the average width of the confidence interval (accuracy), the coverage rate (coverage), and the proportion of times the *p*-value was lower than the significance criterion (power). It requires setting the parameter = "..." argument.
- names: The parameter names in the condition with the least parameters (i.e., parameter names that apply to each experimental condition).

Value

A data.frame.

Examples

```
# Return data frame with number of estimation problems per experimental condition
give(out_preliminary, "estimation_problems")

# Return data frame with performance measures for "wB2~wA1" per experimental condition
give(out_preliminary, "results", parameter = "wB2~wA1")

# Return character vector with parameter names
give(out_preliminary, "names")
```

---

plot.powRICLPM *Plot Results From* powRICLPM *Object*

Description

Visualizes (using **ggplot2**) the results from a powRICLPM analysis, for a specific parameter, across all experimental conditions. By default, sample size is plotted on the x-axis, power on the y-axis, with results colored by the number of time points, wrapped by the proportion of between-unit variance, and shaped by the reliability. Optionally, other variables can be mapped to the y-axis, x-axis, color, shape, and facets.

Usage

```
## S3 method for class 'powRICLPM'
plot(
  x,
  y = "power",
  ...,
  parameter = NULL,
  color_by = "time_points",
  shape_by = "reliability",
  facet_by = "ICC"
)
```

Arguments

x	A powRICLPM object.
y	(optional) A character string, specifying which outcome is plotted on the y-axis (see "Details").
...	(don't use)
parameter	Character string of length 1, denoting the parameter to visualize the results for.
color_by	Character string of length 1, denoting what variable to map to color (see "Details").
shape_by	Character string of length 1, denoting what variable to map to point shapes (see "Details").
facet_by	Character string of length 1, denoting what variable to facet by (see "Details").

Details

Mapping Options: The following outcomes can be plotted on the y-axis:

- `average`: The average estimate.
- `MSE`: The mean square error.
- `coverage`: The coverage rate
- `accuracy`: The average width of the confidence interval.
- `SD`: Standard deviation of parameter estimates.
- `SEAvg`: Average standard error.
- `bias`: The absolute difference between the average estimate and population value.

The following variables can be mapped to color, shape, and facet:

- `sample_size`: Sample size.
- `time_points`: Time points.
- `ICC`: Intraclass correlation (ICC).
- `reliability`: Item-reliablity.

Value

A `ggplot2` object.

See Also

`give`: Extract information (e.g., performance measures) for a specific parameter, across all experimental conditions. This function is used internally by `plot.powRICLPM`.

Examples

```
# Visualize power for "wB2~wA1" across simulation conditions
plot(out_preliminary, parameter = "wB2~wA1")

# Visualize bias for "wB2~wA1" across simulation conditions
plot(out_preliminary, y = "bias", parameter = "wB2~wA1")

# Visualize coverage rate for "wB2~wA1" across simulation conditions
plot(out_preliminary, y = "coverage", parameter = "wB2~wA1")

# Visualize MSE for autoregressive effect across simulation conditions
plot(out_preliminary, y = "MSE", parameter = "wA2~wA1")

# Error: No parameter specified
try(plot(out_preliminary))
```

powRICLPM *Power analysis for the RI-CLPM (and STARTS model)*

Description

Perform a Monte Carlo power analysis for the random intercept cross-lagged panel model (RI-CLPM) and the stable trait autoregressive trait state model (STARTS). This function computes performance metrics such as bias, mean square error, coverage, power, etc, for all model parameters, and can perform power analyses across multiple experimental conditions simultaneously. Conditions are defined in terms of sample size, number of time points, proportion of between-unit variance (ICC), and indicator reliability. See "Details" for information on (a) internal data simulation, (b) internal model estimation, (c) powRICLPM's naming conventions of parameters, (d) parallel execution capabilities for speeding up the analysis, and (e) various extensions, such as the option to include measurement errors for data generation and estimation (i.e., the STARTS model), imposing various constraints over time, and many more.

Usage

```
powRICLPM(
  target_power = 0.8,
  search_lower = NULL,
  search_upper = NULL,
  search_step = 20,
  sample_size = NULL,
  time_points,
  ICC,
  RI_cor,
  Phi,
  within_cor,
  reliability = 1,
  skewness = 0,
  kurtosis = 0,
  estimate_ME = FALSE,
  significance_criterion = 0.05,
  alpha = NULL,
  reps = 20,
  bootstrap_reps = NULL,
  seed = NA,
  constraints = "none",
  bounds = FALSE,
  estimator = "ML",
  save_path = NULL,
  software = "lavaan"
)
```

Arguments

target_power A numeric value between 0 and 1, denoting the targeted power level.

search_lower — A positive integer, denoting the lower bound of a range of sample sizes.

search_upper — A positive integer, denoting the upper bound of a range of sample sizes.

search_step — A positive integer, denoting an increment in sample size.

sample_size — (optional) An integer (vector), indicating specific sample sizes at which to evaluate power, rather than specifying a range using the search_* arguments.

time_points — An integer (vector) with elements at least larger than 3, indicating number of time points.

ICC — A double (vector) with elements between 0 and 1, denoting the proportion of (true score) variance at the between-unit level. When measurement error is included in the data generating model, ICC is computed as the variance of the random intercept factor divided by the true score variance (i.e., controlled for measurement error).

RI_cor — A double between 0 and 1, denoting the correlation between random intercepts.

Phi — A matrix, with standardized autoregressive effects (on the diagonal) and cross-lagged effects (off-diagonal) in the population. Columns represent predictors and rows represent outcomes.

within_cor — A double between 0 and 1, denoting the correlation between the within-unit components.

reliability — (optional) A numeric vector with elements between 0 and 1, denoting the reliability of the variables (see "Details").

skewness — (optional) A numeric, denoting the skewness values for the observed variables (see simulateData).

kurtosis — (optional) A numeric value, denoting the excess kurtosis values (i.e., compared to the kurtosis of a normal distribution) for the observed variables (see simulateData).

estimate_ME — (optional) A logical, denoting if measurement error variance should be estimated in the RI-CLPM (see "Details").

significance_criterion — (optional) A double, denoting the significance criterion.

alpha — (don't use) Deprecated, use significance_criterion instead.

reps — A positive integer, denoting the number of Monte Carlo replications to be used during simulations.

bootstrap_reps — (superseded) Uncertainty regarding simulation estimates is now computed analytically based on Morris et al. (2017). This argument is not used anymore.

seed — An integer of length 1. If multiple cores are used, a seed will be used to generate a full L'Ecuyer-CMRG seed for all cores.

constraints — (optional) A character string, specifying the type of constraints that should be imposed on the estimation model (see "Details").

bounds — (optional) A logical, denoting if bounded estimation should be used for the latent variable variances in the model (see "Details").

estimator — (optional) A character string of length 1, denoting the estimator to be used (default: ML, see "Details").

save_path — A character string of length 1, naming the directory to save (data) files to (used for validation purposes of this package). Variables are saved in alphabetical and numerical order.

software — A character string of length, naming which software to use for simulations; either "lavaan" or "Mplus" (see "Details").

Details

A rationale for the power analysis strategy implemented in this package can be found in Mulder (2023).

Data Generation: Data are generated using simulateData from the **lavaan** package. Based on Phi and within_cor, the residual variances and covariances for the within-components at wave 2 and later are computed, such that the within-components themselves have a variance of 1. This implies that the lagged effects in Phi can be interpreted as standardized effects.

Model Estimation using lavaan: When software = "lavaan" (default), generated data are analyzed using lavaan from the **lavaan** package. The default estimator is maximum likelihood (ML). Other maximum likelihood based estimators implemented in **lavaan** can be specified as well. When skewed or kurtosed data are generated (using the skewness and kurtosis arguments), the estimator defaults to robust maximum likelihood MLR. The population parameter values are used as starting values.

Parameter estimates from non-converged model solutions are discarded from the results. When bounds = FALSE, inadmissible parameter estimates from converged solutions (e.g., a negative random intercept variance) are discarded. When bounds = TRUE, inadmissible parameter estimates are retained following advice by De Jonckere and Rosseel (2022). The results include the minimum estimates for all parameters across replications to diagnose which parameter(s) might be the cause of the inadmissible solution.

Using Mplus: When software = "Mplus", Mplus input files will be generated and saved into save_path. Note that it is not possible to generate skewed or kurtosed data in Mplus via the powRICLPM package. Furthermore, bounded estimation is not available in Mplus. Therefore, the skewness, kurtosis, and bounds will be ignored when software = "Mplus".

Naming Conventions Observed and Latent Variables: The observed variables in the RI-CLPM are given default names, namely capital letters in alphabetical order, with numbers denoting the measurement occasion. For example, for a bivariate RICLPM with 3 time points, we observe A1, A2, A3, B1, B2, and B3. Their within-components are denoted by wA1, wA2, ..., wB3, respectively. The between-components have RI_ prepended to the variable name, resulting in RI_A and RI_B.

Parameters are denoted using **lavaan** model syntax (see the **lavaan** website). For example, the random intercept variances are denoted by RI_A~~RI_A and RI_B~~RI_B, the cross-lagged effects at the first wave as wB2~wA1 and wA2~wB1, and the autoregressive effects as wA2~wA1 and wB2~wB1. Use give(object, "names") to extract parameter names from the powRICLPM object.

Parallel Processing and Progress Bar: To speed up the analysis, power analysis for multiple experimental conditions can be executed in parallel. This has been implemented using **future**.

By default the analysis is executed sequentially (i.e., single-core). Parallel execution (i.e., multi-core) can be setup using plan, for example plan(multisession, workers = 4). For more information and options, see https://future.futureverse.org/articles/future-1-overview.html#controlling-how-futures-are-resolved.

A progress bar displaying the status of the power analysis has been implemented using **progressr**. By default, a simple progress bar will be shown. For more information on how to control this progress bar and several other notification options (e.g., auditory notifications), see https://progressr.futureverse.org.

Extension: Measurement Errors (STARTS model): Including measurement error to the RI-CLPM makes the model equivalent to the bivariate STARTS model by Kenny and Zautra (2001) without constraints over time. Measurement error can be added to the generated data through the reliability argument. Setting the reliability-argument to 0.8 implies that 80 percent is the true score variance, and 20 measurement error variance. ICC then denotes the proportion of *true score variance* captured by the random intercept factors. Estimating measurement errors (i.e., the STARTS model) is done by setting estimate_ME = TRUE.

Extension: Imposing Constraints: The following constraints can be imposed on the estimation model using the constraints = "..." argument:

- lagged: Time-invariant autoregressive and cross-lagged effects.
- residuals: Time-invariant residual variances.
- within: Time-invariant lagged effects and residual variances.
- stationarity: Constraints such that at the within-unit level a stationary process is estimated. This included time-invariant lagged effects, and constraints on the residual variances.
- ME: Time-invariant measurement error variances. Only possible when estimate_ME = TRUE.

Extension: Bounded Estimation: Bounded estimation is useful to avoid nonconvergence in small samples. Here, automatic wide bounds are used as advised by De Jonckere and Rosseel (2022), see optim.bounds in lavOptions. This option can only be used when no constraints are imposed on the estimation model.

Value

An object of class powRICLPM, upon which summary(), print(), and plot() can be used. The returned object is a list with a conditions and session element. condition itself is a list of experimental conditions, where each element is again a list containing the input and output of the power analysis for that particular experimental condition. session is a list containing information common to all experimental conditions.

Author(s)

Jeroen D. Mulder <j.d.mulder@uu.nl>

References

De Jonckere, J., & Rosseel, Y. (2022). Using bounded estimation to avoid nonconvergence in small sample structural equation modeling. *Structural Equation Modeling*, *29*(3), 412-427. doi:10.1080/10705511.2021.1982716

Kenny, D. A., & Zautra, A. (2001). Trait–state models for longitudinal data. *New methods for the analysis of change* (pp. 243–263). American Psychological Association. doi:10.1037/10409008

Mulder, J. D. (2022). Power analysis for the random intercept cross-lagged panel model using the *powRICLPM* R-package. *Structural Equation Modeling*. doi:10.1080/10705511.2022.2122467

See Also

- summary.powRICLPM: Summarize the setup of powRICLPM object.
- give: Extract information from powRICLPM objects.
- plot.powRICLPM: Visualize results powRICLPM object for a specific parameter.

Examples

```
# Define population parameters for lagged effects
Phi <- matrix(c(.4, .1, .2, .3), ncol = 2, byrow = TRUE)

# (optional) Set up parallel computing (i.e., multicore, speeding up the analysis)
library(future)
library(progressr)
future::plan(multisession, workers = 6)

## Not run:
# Run analysis (`reps` is small, because this is an example)
with_progress({
  out_preliminary <- powRICLPM(
    target_power = 0.8,
    search_lower = 500,
    search_upper = 700,
    search_step = 100,
    time_points = c(3, 4),
    ICC = c(0.4, 0.6),
    reliability = c(1, 0.8),
    RI_cor = 0.3,
    Phi = Phi,
    within_cor = 0.3,
    reps = 100,
    seed = 1234
  )
})

## End(Not run)
```

---

print.powRICLPM *Print* powRICLPM *Object*

Description

print.powRICLPM prints a table listing all experimental conditions contained in the powRICLPM object, as well as the frequency of the estimation problems that occurred in each.

Usage

```
## S3 method for class 'powRICLPM'
print(x, ...)
```

Arguments

x	A powRICLPM object.
...	(don't use)

Value

No return value, called for side effects.

---

summary.powRICLPM *Summarize Results from* powRICLPM *Object*

Description

S3 method for class powRICLPM. summary.powRICLPM summarizes the setup and results of the powRICLPM analysis. Depending on the arguments that are set, summary.powRICLPM provides a different summary (see "Details").

Usage

```
## S3 method for class 'powRICLPM'
summary(
  object,
  ...,
  parameter = NULL,
  sample_size = NULL,
  time_points = NULL,
  ICC = NULL,
  reliability = NULL
)
```

Arguments

object	A powRICLPM object.
...	(don't use)
parameter	Character string of length 1 denoting the parameter to visualize the results for.

sample_size	(optional) An integer, denoting the sample size of the experimental condition of interest.
time_points	(optional) An integer, denoting the number of time points of the experimental condition of interest.
ICC	(optional) A double, denoting the proportion of variance at the between-unit level of the experimental condition of interest.
reliability	(optional) An integer, denoting the reliability of the indicators of the experimental condition of interest.

Details

summary.powRICLPM provides a different summary of the powRICLPM object, depending on the additional arguments that are set:

- When sample_size = ..., time_points = ..., ICC = ..., and reliability are set: Estimation information and results for all parameters across experimental conditions.
- When parameter = "..." is set: Estimation information and results for a specific parameter across all experimental conditions.
- No additional arguments: Characteristics of the different experimental conditions are summarized, as well as session info (information that applies to all conditions, such the number of replications, etc.).

Interpretation Output: Depending on the arguments that you set, summary() prints a table with different analysis outcomes in the columns and where each row refers to a different experimental condition. The following information is available:

- Sample size, Time points, ICC, Reliability: The experimental condition that the row refers to.
- Population: The true value of the parameter.
- Avg: The average (across replications) parameter estimate.
- Bias: The difference between the population value and the average parameter estimate.
- Min: The lowest (across replications) parameter estimate.
- SD: The standard deviation of the parameter estimate over replications.
- SEAvg: The average (across replications) standard error of the parameter estimate.
- MSE: The parameter mean square error, combining a parameter's bias and efficiency.
- Accuracy: The average (across replications) width of the confidence interval.
- Cover: The coverage rate, representing the proportion of times (across replications) the true parameter estimate fell in the confidence interval.
- Power: The proportion of times (across replications) the confidence interval did not contain zero.
- Error: The number of replications that failed to run (i.e., lavaan() produced an error).
- Not converged: The number of replications that did not converge to a solution.
- Inadmissible: The number of replications that converged to an inadmissible solution (e.g., a variance estimated to be lower than zero).

Value

No return value, called for side effects.

Examples

```
# Get setup of powRICLPM analysis and convergence issues
summary(out_preliminary)

# Performance measures for "wB2~wA1" parameter across experimental conditions
summary(out_preliminary, parameter = "wB2~wA1")

# Performance measures for all parameters, for specific experimental condition
summary(out_preliminary, sample_size = 700, time_points = 4, ICC = .3, reliability = 1)
```

Index