Package: POEMED (via r-universe)

October 19, 2026

Type Package

Title Power Enhancement for High-Dimensional Mediation Analysis

Version 1.0.0

Date 2026-10-05

Description Provides powerful global tests for high-dimensional mediation analysis with heterogeneous (and even contrasting) mediation effects, for continuous, binary, and count outcomes. The power enhancement (PE) idea augments a Wald test on the total indirect effect with a component that captures marginal signals from individual mediators, so that active mediators are detected even when their individual indirect effects cancel and the total indirect effect is zero. The package also identifies which individual mediators are active with familywise error rate or false discovery rate control. The methods implement Yu and Kelley (in press), ``Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis'' (Journal of the American Statistical Association); the package also simulates mediation data under the homogeneous, heterogeneous, and contrasting patterns that the article studies, ships its empirical health-expenditure data as a benchmark and reruns that empirical analysis, and runs the designs of its Monte Carlo size and power studies. The name, 'POEMED', from POwer-Enhanced MEDiation, encodes the method. Every estimation and testing function returns a tidy 'data.frame' that prints with sensible rounding and composes with the broader R ecosystem.

Imports generics, glmnet, graphics, ncvreg, parallel, stats, utils, withr (>= 3.0.3)

Suggests knitr, rmarkdown, testthat (>= 3.2.0), tools

License GPL (>= 3)

Copyright Xiufan Yu and Ken Kelley, except the WHO_health_mediation and WHO_indicator_codebook data sets, which redistribute World Health Organization (Global Health Expenditure Database) and World Bank (World Development Indicators) data under the terms stated on the help page of WHO_health_mediation and, in the

source package, in the file LICENSE.note.

URL https://github.com/yelleKneK/POEMED, https://yelleknek.github.io/POEMED/

BugReports https://github.com/yelleKneK/POEMED/issues

Encoding UTF-8

Language en-US

LazyData true

LazyDataCompression xz

VignetteBuilder knitr

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

Depends R (>= 4.1.0)

Config/roxygen2/version 8.1.0

Repository https://yelleknek.r-universe.dev

Date/Publication 2026-10-07 12:12:20 UTC

RemoteUrl https://github.com/yelleknek/poemed

RemoteRef HEAD

RemoteSha 9e78f8a822ae4afe38e7982cdbd99a615da26100

Contents

example_data . . . 3
guo_calibration . . . 4
mediation_ar1_cov . . . 6
pe_identification_study . . . 7
pe_mediate . . . 10
pe_mediation . . . 12
pe_mediation_linear . . . 18
pe_mediation_logistic . . . 23
pe_mediation_poisson . . . 26
pe_mediators . . . 30
pe_power_curve . . . 32
pe_selection . . . 35
pe_simulation_study . . . 36
plot.poemed_tbl . . . 38
poemed_broom . . . 39
poemed_tbl . . . 40
simulate_guo_mediation . . . 42
simulate_mediation_data . . . 44
ss_power_pe_mediation . . . 47
summary.poemed_tbl . . . 50
WHO_health_mediation . . . 51

WHO_indicator_codebook 54
WHO_mediation_analysis 55
WHO_mediation_design 59

Index **62**

`example_data` *Example High-Dimensional Mediation Data Sets*

Description

Three small simulated data sets, one per outcome type, for trying the POEMED tests and reproducing the help-page examples. Each was generated by `simulate_mediation_data()` under a homogeneous mediation pattern (a few same-sign active mediators among many null ones) with three confounders, at seed 113, so the generating truth is known and travels with the data.

Usage

```
example_continuous
example_binary
example_count
```

Format

Each is a list, the return value of `simulate_mediation_data()`, with components:

X Exposure matrix, $n \times 1$.

M Candidate mediator matrix, $n \times 100$.

Y Outcome vector of length n: continuous for `example_continuous`, 0/1 for `example_binary`, nonnegative counts for `example_count`.

Z Confounder matrix, $n \times 3$.

alpha_m, Gamma_x, alpha_x The mediator-on-outcome, exposure-on-mediator, and direct-effect coefficients used to generate the data.

alpha_z, Gamma_z The confounder coefficients the simulator drew from U(-0.5, 0.5) and used: `alpha_z`, the three confounder-on-outcome coefficients, and `Gamma_z`, the 3×100 matrix of confounder-on-mediator coefficients, in which each confounder loads equally on every mediator (each row holds one value).

beta The total indirect effect $\Gamma_x \alpha_m$: 0.42 for `example_continuous`, 0.8 for `example_binary`, and 0.24 for `example_count`.

active_mediators Indices of the truly active mediators: mediators 1 to 5 in `example_continuous`, and mediators 1, 2, and 5 in `example_binary` and `example_count`.

n, p, q, outcome, pattern The generating settings.

The sample sizes are $n = 200$ (`example_continuous`) and $n = 300$ (`example_binary`, `example_count`).

In `example_count` the simulator's overflow guard fired (the largest log-mean was 5.88; see the Value section of `simulate_mediation_data()`), so the counts follow a log-mean whose coefficients are the recorded `alpha_m`, `alpha_x`, and `alpha_z` multiplied by 0.71 (with an intercept of 0.83), and a total indirect effect of 0.17 where `beta` records 0.24.

Source

Simulated with simulate_mediation_data(); see data-raw/example_data.R in the package sources.

See Also

pe_mediation(), simulate_mediation_data().

Examples

```
data(example_continuous)
str(example_continuous$active_mediators)   # truly active: mediators 1 to 5
fit <- pe_mediation(example_continuous$X, example_continuous$Y,
                    example_continuous$M, Z = example_continuous$Z,
                    outcome = "continuous")
# Both tests reject at the 0.05 level, and the screen identifies one of
# the five truly active mediators (mediator 4).
fit
```

---

guo_calibration — *Calibration Constants for the Real-Data-Motivated Heterogeneous Setting*

Description

The fixed constants for the real-data-motivated heterogeneous mediation simulation of Yu and Kelley (in press), calibrated to the DNA-methylation case study of Guo et al. (2022): a $p = 1008$-mediator linear model with eleven active loci whose effects mix positive and negative signs. simulate_guo_mediation() generates data from these constants; the object is exported so the exact calibration is inspectable.

Usage

```
guo_calibration
```

Format

A list with components:

p, s, n The mediator count (1008), confounder count (9, an intercept plus eight covariates), and case-study sample size (85).

locations Indices of the eleven active mediators.

alpha_m Outcome-mediator coefficients at the active loci (the designed simulation values).

Gamma_x Exposure-mediator coefficients at the active loci.

alpha_z Confounder-outcome coefficients (length 9).

Gamma_z Confounder-mediator coefficients at the active loci (an 11 by 9 matrix).

beta_per_c1 The total indirect effect per unit `c1`, equal to `sum(Gamma_x * alpha_m)` ≈ -1.597 (the article reports -1.5977 from the unrounded coefficients).

alpha_m_estimated The raw Guo et al. (2022) estimate of the outcome-mediator coefficients (for reference; the simulation uses the designed `alpha_m`).

alpha_m_variants Two alternative outcome-mediator coefficient sets the article also studies, `homogeneous_like` and `contrasting_like`, which remain heterogeneous when paired with `Gamma_x`.

Details

Pairing the exposure-mediator coefficients `Gamma_x` with the outcome-mediator coefficients `alpha_m` gives a total indirect effect of `beta_per_c1` ≈ -1.597 per unit of the signal scale c_1. (The article reports -1.5977, computed from the full-precision Guo coefficients; the values shipped here are those coefficients rounded to the three decimals printed in the supplement, which give -1.597.)

Source

Transcribed from the supplement of Yu and Kelley (in press) (its parameter-configuration section), which in turn calibrates to the `simulation_allS.Rdata` of Guo et al. (2022).

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Guo, X., Li, R., Liu, J., & Zeng, M. (2022). High-dimensional mediation analysis for selecting DNA methylation loci mediating childhood trauma and cortisol stress reactivity. *Journal of the American Statistical Association, 117*(539), 1110–1121. doi:10.1080/01621459.2022.2053136

See Also

`simulate_guo_mediation()`

Other mediation simulation: `mediation_ar1_cov()`, `pe_identification_study()`, `pe_power_curve()`, `pe_simulation_study()`, `plot.poemed_tbl()`, `simulate_guo_mediation()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
str(guo_calibration, max.level = 1)
# The eleven individual indirect effects: nine negative and two positive
sign(guo_calibration$Gamma_x * guo_calibration$alpha_m)
# Their sum is the total indirect effect per unit of c1
sum(guo_calibration$Gamma_x * guo_calibration$alpha_m)
```

`mediation_ar1_cov` *Autoregressive (AR(1)) Covariance Matrix for Mediator Simulation*

Description

Builds the $p \times p$ first-order autoregressive covariance matrix $\Sigma = (\rho^{|i-j|})_{i,j}$, the intercorrelation structure imposed on the mediator noise throughout the article's Monte Carlo studies. It is exported as a small reusable utility so a user can inspect or reuse the exact covariance the simulations use.

Usage

```
mediation_ar1_cov(p, rho = 0.5)
```

Arguments

`p`	Number of mediators (matrix dimension); a positive integer.
`rho`	Autocorrelation parameter in $(-1, 1)$. The covariance between mediators i and j is $\rho^{|i-j|}$, so nearby mediators are more strongly correlated and the correlation decays geometrically with separation. Default 0.5, the value used in the article.

Value

A $p \times p$ numeric matrix with ones on the diagonal and $\rho^{|i-j|}$ off the diagonal.

Author(s)

Xiufan Yu and Ken Kelley

See Also

`simulate_mediation_data()`, which uses this covariance to draw correlated mediator noise.

Other mediation simulation: `guo_calibration`, `pe_identification_study()`, `pe_power_curve()`, `pe_simulation_study()`, `plot.poemed_tbl()`, `simulate_guo_mediation()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
mediation_ar1_cov(4, rho = 0.5)
```

pe_identification_study

Monte Carlo Study of Individual-Mediator Identification (FWER / FDR)

Description

Evaluates how well the power-enhanced screen recovers the *individual* truly active mediators, the experiment reported in the article's supplement. For each signal-strength scale c_1 on a grid it simulates many data sets, identifies a set of active mediators under each multiplicity method, and scores the identified sets against the known truth, returning the empirical familywise error rate, false discovery rate, precision, and recall. This complements `pe_power_curve()`, which evaluates the *global* test rather than which mediators are flagged.

Usage

```
pe_identification_study(
  n,
  p,
  outcome = c("continuous", "binary", "count"),
  pattern = c("homogeneous", "contrasting", "heterogeneous"),
  c1_grid = seq(0, 1, by = 0.25),
  c2 = 0.5,
  n_rep = 200,
  methods = c("Bonferroni", "BH", "BY"),
  error_level = 0.05,
  truth = c("outcome_effect", "mediation"),
  lambda_grid = seq(0.05, 10, length.out = 100),
  outcome_args = list(),
  cores = 1L,
  seed = NULL,
  progress = FALSE
)
```

Arguments

`n, p`	Number of observations and candidate mediators per data set.
`outcome`	Outcome type passed to `simulate_mediation_data()`: `"continuous"`, `"binary"`, or `"count"`.
`pattern`	Mediation pattern: `"homogeneous"` or `"contrasting"` (`"heterogeneous"` is a synonym for `"contrasting"`).
`c1_grid`	Numeric vector of signal-strength scales to sweep. Default `seq(0, 1, by = 0.25)`. Include 0 to estimate the null familywise error rate.
`c2`	Direct effect used in the simulation. Default 0.5.
`n_rep`	Number of Monte Carlo replications per grid point. Default 200. The article uses 1000.

`methods` Multiplicity methods to evaluate, one or more of `"Bonferroni"` (familywise error rate), `"BH"`, and `"BY"` (false discovery rate), each named once. Default all three.

`error_level` Target error rate for the mediator-screening step (the FWER level for Bonferroni, the FDR level for BH and BY). Default 0.05.

`truth` Which mediators count as truly active: `"outcome_effect"` (nonzero outcome coefficient, the convention of the supplement's identification tables) or `"mediation"` (both paths nonzero, the article's definition of the true active set in its Theorem 3). See Details.

`lambda_grid` Passed to `pe_mediation()`: the tuning grid for the penalized fit, by default `seq(0.05, 10, length.out = 100)` (see Details). A shorter or narrower grid speeds a study up.

`outcome_args` A named list of further arguments forwarded to `simulate_mediation_data()` (for example `rho`, `q`, `d`, `tau`, `alpha_m`). Default empty. With `d` above 0 and no `alpha_z` or `Gamma_z`, the simulator draws the confounder coefficients afresh in every replication. Each name must identify one argument of `simulate_mediation_data()`; an unknown or ambiguous name stops with an error. The design arguments this function sets itself (`n`, `p`, `outcome`, `pattern`, `c1`, `c2`) and `seed` may not appear here, whether spelled in full or abbreviated.

`cores` Number of CPU cores for the replications. Values above 1 fork via the base **parallel** package (Unix only) and run the replications on the `"L'Ecuyer-CMRG"` generator (the caller's generator kind is restored on exit); a seeded parallel run is reproducible across runs at the same `cores` but need not match a serial run. The parallel streams are handed out afresh from the same generator state at every point of `c1_grid`, so with `cores` above 1 each grid point replays the same `n_rep` random draws (common random numbers) and the rows at different grid points share their Monte Carlo noise. With `cores = 1` every grid point draws fresh data. Default 1.

`seed` Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. `NULL` (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

`progress` `TRUE` or `FALSE`; if `TRUE`, print a line per grid point. Default `FALSE`.

Details

The supplement of Yu and Kelley (in press) reports four metrics, empirical FWER, empirical FDR, precision, and recall, without defining them; this function computes them as follows. In each replication the identified active set is compared with the truth set. The familywise error rate is the proportion of replications identifying at least one mediator outside the truth set. The false discovery rate is the mean false discovery proportion (zero when nothing is identified). Precision is the mean proportion of identified mediators that are true (scored zero when nothing is identified), and recall is the mean proportion of true mediators identified. The article uses 1000 replications.

The familywise error rate is a proportion of replications, so its Monte Carlo standard error is $\sqrt{r(1-r)/n_{valid}}$, with r the reported rate. The other three metrics are means of per-replication fractions between 0 and 1, for which that formula is only an upper bound. Their standard error is

the standard deviation of the per-replication values divided by $\sqrt{n_{valid}}$, and for recall it can be as little as a third of the bound.

What counts as a true mediator is set by `truth`. Under `truth = "outcome_effect"` (the default) a mediator is true when its outcome coefficient $\alpha_{m,j}$ is nonzero. This is the convention of the supplement's identification tables: their $c_1 = 0$ rows report positive precision and recall, which only a truth set that is nonempty when the exposure-on-mediator paths are all zero allows. There the familywise error rate is the probability of identifying as active a mediator with no outcome effect. Under `truth = "mediation"` a mediator is true only when both paths are nonzero. That is the definition of the true active set in Theorem 3 of the article, against which its familywise error guarantee is stated, and of the simulator's `active_mediators`. At $c_1 = 0$ no mediator is then truly active, any identification is a false positive, and recall is `NA`. For nonzero c_1 the two definitions agree under the article's designs, whose exposure-on-mediator loadings are all nonzero.

The tuning grid governs these rates as much as the screen does. Every replication searches the same grid: the package default `seq(0.05, 10, length.out = 100)`, or the grid passed as `lambda_grid`, and each fit keeps the best value the grid offers. A shorter or narrower grid speeds a study up; the single-fit warning for a choice at an end of the grid is not raised inside a study, which searches a fixed grid by design.

Value

A tidy `data.frame` of class `poemed_tbl` with one row per (c_1, method) combination and columns `c1`, `method`, `fwer`, `fdr`, `precision`, `recall` (as defined in Details), `n_valid` (replications whose fit succeeded), and `n_empty` (replications whose penalized fit selected no mediator). The settings are recorded in the attributes `outcome`, `pattern`, `n`, `p`, `n_rep`, `error_level`, `truth`, and `lambda_grid`.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

`pe_power_curve()` for the global test, `pe_selection()` for the per-method identified active set of a single fit.

Other mediation simulation: `guo_calibration`, `mediation_ar1_cov()`, `pe_power_curve()`, `pe_simulation_study()`, `plot.poemed_tbl()`, `simulate_guo_mediation()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
# n_rep = 5 keeps this example fast; a rate from five replications has a
# Monte Carlo standard error of up to 0.22, so read the shape, not the
# numbers. A reported study uses the article's design (n = 300, p = 500)
# and 1000 replications. A study is also where a shorter tuning grid
# pays: every replication searches the whole grid, so this one passes 20
# values from 0.05 to 2 instead of the default 100 values from 0.05 to 10
# (see the lambda_grid argument of ?pe_mediation).
set.seed(113)
pe_identification_study(n = 200, p = 60, outcome = "continuous",
                        pattern = "contrasting", c1_grid = c(0, 1),
                        n_rep = 5,
                        lambda_grid = seq(0.05, 2, length.out = 20))
```

`pe_mediate` *Power-Enhanced Mediation Test From a Data Frame (Formula Interface)*

Description

A convenience front end to `pe_mediation()` for users who have a data frame rather than ready-made matrices. Give it the outcome and exposure as a formula, name the mediator columns, and (optionally) name confounder columns; it assembles the exposure, outcome, mediator, and confounder matrices (turning factor confounders into indicator variables) and runs the test. This is the gentler entry point; the matrix interface `pe_mediation()` is the workhorse.

Usage

```
pe_mediate(
  formula,
  data,
  mediators,
  confounders = NULL,
  outcome = c("continuous", "binary", "count"),
  ...
)
```

Arguments

`formula`	A two-sided formula `outcome ~ exposure` naming the outcome and one or more exposure columns in `data`. A `.` stands for every other column of `data`, so either name the exposures explicitly or subtract the mediator and confounder columns (`y ~ . - m1 - m2`); a `.` that brings a mediator or confounder in as an exposure stops.
`data`	A `data.frame` containing the outcome, exposure, mediator, and any confounder columns.

`mediators` Character vector of mediator column names in `data` (the candidate mediators `M`). At least two columns, each listed once, and none of them the outcome, an exposure, or a confounder.

`confounders` Optional character vector of confounder column names. Factor and character columns are expanded into indicator variables; numeric columns enter as is. Default `NULL`; an empty vector, `character(0)`, also means no confounders.

`outcome` Outcome type: `"continuous"`, `"binary"`, or `"count"`.

`...` Further arguments passed to `pe_mediation()` (for example `method`, `error_level`, `report_all_methods`). Among them is `lambda_grid`, the grid of candidate tuning parameters that HBIC searches, `seq(0.05, 10, length.out = 100)` by default for every outcome family: a shorter or narrower grid makes the fit faster, and a wider or finer one is worth trying when the value chosen is the smallest or largest in the grid (a non-empty fit warns, and the print footer, which reports the full model's choice, says so). See the argument on `pe_mediation()`. The reduced-model grid `lambda_grid_reduced` is an argument of `pe_mediation_linear()` alone and cannot be passed here; for a continuous outcome the reduced model searches `lambda_grid`.

Details

Each column plays one role. The outcome and the exposures come from `formula`, the mediators from `mediators`, and the confounders from `confounders`. A column listed in two roles, or a mediator listed twice, stops with a message naming it: the outcome listed as a mediator, for example, would be "identified" as its own mediator.

Missing values follow one complete-case rule across every column the model uses (outcome, exposures, mediators, and confounders). A row with a missing value in any of them is dropped, and a message reports how many rows were dropped and which columns held the missing values. The method has no missing-data mechanism, so the test is computed on the complete cases; impute beforehand if dropping rows is not appropriate for your data. The matrix interface `pe_mediation()` drops nothing: it stops on a missing value and leaves the choice of complete cases to the user.

The outcome must be numeric (or logical, for a binary outcome). A factor or character outcome stops with a message, because a factor's numeric codes are not its values: a count stored as a factor would otherwise be analyzed as a different count.

Value

The tidy `poemed_tbl` returned by `pe_mediation()`.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

See Also

pe_mediation() for the matrix interface, WHO_mediation_design() for the worked health-expenditure design.

Other mediation tests: pe_mediation(), pe_mediation_linear(), pe_mediation_logistic(), pe_mediation_poisson(), pe_mediators(), pe_selection(), summary.poemed_tbl()

Examples

```
# Build a small data frame and test through the formula interface.
set.seed(113)
d <- simulate_mediation_data(n = 200, p = 60, outcome = "continuous",
                             pattern = "contrasting", c1 = 1)
df <- data.frame(y = d$Y, x = d$X[, 1], d$M)
meds <- grep("^X", names(df), value = TRUE)   # the mediator columns
# The mediator columns are named, so the footer names the active
# mediators identified (X2, X3, and X4).
pe_mediate(y ~ x, data = df, mediators = meds, outcome = "continuous")
# HBIC chose the smallest value of the default grid here, so the fit
# warned; see the lambda_grid argument of ?pe_mediation.
```

---

pe_mediation *Power-Enhanced Test for High-Dimensional Mediation*

Description

Tests whether any mediator is active among a large, possibly intercorrelated set of candidate mediators, for a continuous, binary, or count outcome, with the power-enhanced (PE) test of Yu and Kelley (in press). This is the main entry point of **POEMED**: name the outcome type and it dispatches to the appropriate model, returning one tidy table that reports the benchmark Wald test on the total indirect effect side by side with the PE test, plus the individual mediators the screen identified as active.

Usage

```
pe_mediation(
  X,
  Y,
  M,
  Z = NULL,
  outcome = c("continuous", "binary", "count"),
  method = c("Bonferroni", "BH", "BY"),
  scale = TRUE,
  error_level = 0.05,
  report_all_methods = FALSE,
  drop_constant = FALSE,
  lambda_grid = seq(0.05, 10, length.out = 100)
)
```

Arguments

`X` Numeric matrix of exposures with n rows and q columns (q is usually 1). A numeric vector is treated as a single exposure.

`Y` Numeric outcome vector of length n. Continuous for `outcome = "continuous"`, coded 0/1 for `"binary"`, and nonnegative integer counts for `"count"`.

`M` Numeric matrix of candidate mediators with n rows and p columns; p may exceed n. Column names, when present, label the mediators in the print footer and the mediator table; without them mediators are reported by column position.

`Z` Optional numeric matrix of confounders with n rows. Default `NULL` (no confounders).

`outcome` Outcome type: `"continuous"` (linear model), `"binary"` (logistic model), or `"count"` (Poisson log-link model).

`method` Multiplicity adjustment used to screen individual mediators in the PE component: `"Bonferroni"` (familywise error rate control, the default), `"BH"`, or `"BY"` (false discovery rate control by the procedures of Benjamini and Hochberg, 1995, and of Benjamini and Yekutieli, 2001; `"BH"` assumes independence or positive dependence among mediators, `"BY"` is valid under arbitrary dependence). See Details for the screening threshold each one uses.

`scale` Logical; if `TRUE` (default) the exposures, mediators, and confounders are each standardized to mean 0 and standard deviation 1 before fitting, as in the article, and a continuous outcome is centered but not rescaled. The total indirect effect is then the change in the outcome per standard deviation of each exposure: in the units of `Y` for a continuous outcome, in log odds for a binary outcome, and in log mean count for a count outcome. With `FALSE` the inputs are used exactly as supplied, and the total indirect effect is per unit of each exposure. The continuous-outcome fit has no intercept, and the default tuning grid was chosen for standardized columns, so `FALSE` is meant for inputs that are already standardized (and, for a continuous outcome, a `Y` that is already centered); on raw inputs it fits a different model and can even change the sign of the estimate. Because a continuous `Y` is not rescaled, the tuning grid is not expressed in the outcome's units, so the active mediators identified, both p-values, and the verdict depend on the units of `Y`: the same data with `Y` multiplied by 10 can identify a different set of mediators.

`error_level` Target error rate for the mediator-screening step, in $(0, 1)$. Interpreted as the familywise error rate when `method = "Bonferroni"` and as the false discovery rate when `method = "BH"` or `"BY"`. Default 0.05. This is distinct from the significance level used to test the global null (which is the user's choice when reading `pval_pe`). The PE test is asymptotically valid for any fixed value in $(0, 1)$, but in finite samples a larger value lets the screen pass more spurious mediators under the null and raises the PE test's size further above the benchmark's (see Details).

`report_all_methods` Logical; if `TRUE`, the identified active set is computed for all three multiplicity methods (Bonferroni, BH, BY) off the same fit and recorded for comparison, retrievable with `pe_selection()`. The `method` argument still drives the primary result. Default `FALSE`.

`drop_constant` Logical; how to handle constant (zero-variance) mediator columns, which cannot be standardized or carry signal. If `FALSE` (the default) they are an error; if `TRUE` they are dropped with a warning and the remaining mediators are renumbered back to their original column positions in the identified active set. Useful for small subgroups where some mediators happen to be constant.

`lambda_grid` Numeric vector of candidate tuning parameters for the SCAD penalty (Fan and Li, 2001) in the penalized mediator fit, which for a binary outcome is adaptively rescaled (see `pe_mediation_logistic()`). The fit is repeated at each value and the one minimizing the high-dimensional BIC (HBIC) of Wang, Kim, and Li (2013) is kept: whatever the grid, the value selected is the best of the values offered, since HBIC is minimized over the grid and not over every positive lambda. The default, `seq(0.05, 10, length.out = 100)` for every outcome family, is written out in the signature so that it can be edited in place. Every value is fit (the search is not adaptive), so the cost of a fit grows with the length of the grid. The grid can be tuned to the situation. Fewer values, or a range narrowed to where earlier fits put the HBIC minimum, make each fit faster, which matters when many fits are run (a simulation study, for example); a wider or finer grid is worth trying when the selected value is the smallest or largest value of the grid, in which case a non-empty fit warns (class `poemed_grid_boundary`) and the print footer, which reports the full model's choice, says so. A continuous outcome also fits a reduced model, the outcome on the mediators and confounders without the exposure, tuned on the same grid by default (see `lambda_grid_reduced` in `pe_mediation_linear()`); its choice is `lambda_selected_reduced` in the `"tuning"` attribute, and the warning covers it as well, naming whether the full model, the reduced model, or both sat at the end. For a continuous outcome the grid is fixed while `Y` is centered but not rescaled, so the value HBIC chooses, and with it the active mediators identified, depends on the units of `Y` (see `scale`). The grid searched and the value chosen are reported in the `"tuning"` attribute and the print footer.

Details

Standard high-dimensional mediation tests target the *total* indirect effect $\beta = \Gamma_x \alpha_m$, the sum of every mediator's individual indirect effect. When some indirect effects are positive and others negative they can cancel, making $\beta = 0$ even though active mediators exist, and a test built on β is then powerless. The benchmark reported here is such a test, the Wald test S_n on β. For a continuous outcome it is the test of Guo et al. (2022), which extends the partially penalized Wald test of Guo et al. (2023) to observed confounders. For binary and count outcomes it is the test of Guo et al. (2024).

The PE test of Yu and Kelley (in press) follows the power enhancement principle of Fan, Liao, and Yao (2015). It adds to S_n the component (equation 2.9 of the article)

$$J_m = \sqrt{p} \sum_{i=1}^{q} \sum_{j \in \hat{S}} \left| \frac{\hat{\alpha}_{m,j}}{\hat{\sigma}_{m,j}} \right| \left| \frac{\hat{\Gamma}_{x,i,j}}{\hat{\sigma}_{\Gamma,i,j}} \right| \mathbf{1} \left\{ \max(p_{1,j}, p_{2,i,j}) < \frac{\alpha_{\mathrm{lvl}}}{sq \log \log n} \right\},$$

a sum over the s penalized-selected candidates $\hat{S}$ of the product of the (standardized) mediator-on-outcome and exposure-on-mediator statistics, kept only for pairs where both paths are individually significant; α_{lvl} is `error_level`. That indicator is the Bonferroni screen used by `method =`

"Bonferroni". Under method = "BH" or "BY" the indicator is instead $\mathbf{1}\{\tilde{p}_{i,j} < \alpha_{\mathrm{lvl}}/\log\log n\}$, where $\tilde{p}_{i,j}$ is the Benjamini and Hochberg (1995) or Benjamini and Yekutieli (2001) adjusted value of $\max(p_{1,j}, p_{2,i,j})$ over the sq pairs (equations S.22 to S.25 of the article's supplement). Because J_m accumulates magnitudes, opposite-signed indirect effects reinforce rather than cancel, so the test $M_{PE} = S_n + J_m$ stays powerful under heterogeneous and contrasting mediation. The indicator also identifies individual active mediators, with asymptotic familywise error rate control under method = "Bonferroni" or false discovery rate control under "BH" / "BY". In this documentation an *active* mediator is one whose effect is nonzero (the article's usage, as in the global null of no active mediator). The screen identifies active mediators, the candidates it finds statistically significant under the multiplicity method, and the fit reports them as its active mediators (the "active_mediators" attribute): the identified set is the method's estimate of the truly active set. *Selected* refers to the first stage, that is, to the candidates the penalized fit keeps (the penalized-selected candidates $\hat{S}$) and to the tuning parameter HBIC selects.

Both statistics are referred to a chi-square distribution with q degrees of freedom. Under the global null of no active mediator, J_m is zero with probability tending to one, so M_{PE} has the same chi-square limit as S_n and the PE test is asymptotically valid for any fixed error_level, by Theorem 1 of Yu and Kelley (in press). That guarantee is asymptotic. In finite samples, under a null in which some mediators affect the outcome but the exposure affects none of them, the screen passes one of those mediators with probability of about error_level / log(log(n)), which is 0.030 at $n = 200$ with error_level = 0.05, and J_m is then large enough to reject. The PE test's size can therefore exceed the benchmark's by up to about that amount. For example, pe_power_curve(200, 100, pattern = "homogeneous", c1_grid = 0, n_rep = 2000, seed = 113) gave rejection rates at the 0.05 level of 0.0475 for the benchmark and 0.0620 for the PE test over 2,000 null samples, and 0.0510 for the PE test with error_level = 0.01 (Monte Carlo standard errors about 0.005). A smaller error_level narrows the gap, at some cost in identifying active mediators.

Value

A tidy data.frame of class poemed_tbl. See pe_mediation_linear() for the row schema and the attributes. The active mediators the screen identified (see Details for how they relate to the truly active mediators and to the penalized-selected candidates) are in the "active_mediators" attribute and the print footer; the tuning parameter HBIC chose is in the "tuning" attribute and the footer. The total indirect effect is on the scale described under scale.

When to Use POEMED

POEMED is for the global question, "is there any active mediator among many candidates?", followed by the identification of individual active mediators. It assumes a linear or generalized linear outcome model, a sparse set of active mediators, standardized inputs, and complete data. Some practical limits, seen on simulated data:

- Sample size. The penalized selection needs n large relative to $\log p$ and to the signal. When it is not, as at $n = 100$ with $p = 2000$ under the contrasting pattern, the selection typically keeps one or two mediators, the screen identifies one of them or none, and the test has little left to draw on.
- Strongly correlated mediators. Correlation among the mediators weakens both the selection and the screen: the screen can come up empty, and the identified active set varies from sample to sample. An autoregressive correlation of 0.9 among neighboring mediators is enough to see this; 0.5 is not.

- Active mediators identified in finite samples. The screen's control of its error rate is asymptotic, so the identified active set can hold a mediator that is not truly active, the more so the stronger the signal. The rate depends on the tuning grid, and a grid that leaves out the smallest values of lambda lowers it at the cost of recall.
- Rare binary outcomes. With only a few percent events the penalized logistic fit that HBIC selects is empty, and the function says so.
- Overdispersed counts. The count model is Poisson. Under negative binomial counts its size held up, but a share of the fits came back empty. A count fit can also take several times as long as a continuous fit of the same design.
- Missing values are not handled; supply complete cases.

For a single mediator, or a few mediators fit as one structural model (indirect effects with confidence intervals, likelihood ratio tests of arbitrary indirect effects, moderated mediation), the **DMAR** package is the tool; POEMED's contribution is the high-dimensional global test. The other high-dimensional methods the article compares against are HILMA (Zhou, Wang, & Zhao, 2020), GlobalTest (Djordjilovic et al., 2019), HDMT (Dai, Stanford, & LeBlanc, 2022), and DACT (Liu et al., 2022). POEMED does not depend on them. HDMT is on CRAN, and GlobalTest is the Bioconductor package **globaltest**. The CRAN package **freebird**, which implemented HILMA, was archived in July 2026, and the CRAN package named **DACT** is an unrelated clinical-trials package, not the method of Liu et al. The package implementing the DACT method of Liu et al. is also named **DACT**, but it is not on CRAN and is instead available through their GitHub repository. The vignette `poemed-vs-competitors` shows how to call HILMA and GlobalTest beside a POEMED fit.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Fan, J., Liao, Y., & Yao, J. (2015). Power enhancement in high-dimensional cross-sectional tests. *Econometrica, 83*(4), 1497–1541. doi:10.3982/ECTA12749

Guo, X., Li, R., Liu, J., & Zeng, M. (2022). High-dimensional mediation analysis for selecting DNA methylation loci mediating childhood trauma and cortisol stress reactivity. *Journal of the American Statistical Association, 117*(539), 1110–1121. doi:10.1080/01621459.2022.2053136

Guo, X., Li, R., Liu, J., & Zeng, M. (2023). Statistical inference for linear mediation models with high-dimensional mediators and application to studying stock reaction to COVID-19 pandemic. *Journal of Econometrics, 235*(1), 166–179. doi:10.1016/j.jeconom.2022.03.001

Guo, X., Li, R., Liu, J., & Zeng, M. (2024). Estimations and tests for generalized mediation models with high-dimensional potential mediators. *Journal of Business & Economic Statistics, 42*(1), 243–256. doi:10.1080/07350015.2023.2174548

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association, 96*(456), 1348–1360. doi:10.1198/016214501753382273

Wang, L., Kim, Y., & Li, R. (2013). Calibrating nonconvex penalized regression in ultra-high dimension. *The Annals of Statistics, 41*(5), 2505–2536. doi:10.1214/13AOS1159

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B, 57*(1), 289–300. doi:10.1111/j.25176161.1995.tb02031.x

Benjamini, Y., & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics, 29*(4), 1165–1188. doi:10.1214/aos/1013699998

Zhou, R. R., Wang, L., & Zhao, S. D. (2020). Estimation and inference for the indirect effect in high-dimensional linear mediation models. *Biometrika, 107*(3), 573–589. doi:10.1093/biomet/asaa016

Djordjilovic, V., Page, C. M., Gran, J. M., Nost, T. H., Sandanger, T. M., Veierod, M. B., & Thoresen, M. (2019). Global test for high-dimensional mediation: Testing groups of potential mediators. *Statistics in Medicine, 38*(18), 3346–3360. doi:10.1002/sim.8199

Dai, J. Y., Stanford, J. L., & LeBlanc, M. (2022). A multiple-testing procedure for high-dimensional mediation hypotheses. *Journal of the American Statistical Association, 117*(537), 198–213. doi:10.1080/01621459.2020.1765785

Liu, Z., Shen, J., Barfield, R., Schwartz, J., Baccarelli, A. A., & Lin, X. (2022). Large-scale hypothesis testing for causal mediation effects with applications in genome-wide epigenetic studies. *Journal of the American Statistical Association, 117*(537), 67–81. doi:10.1080/01621459.2021.1914634

See Also

The outcome-specific workers `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`; `simulate_mediation_data()` to generate data; `pe_power_curve()` to run the designs of the article's size and power studies.

Other mediation tests: `pe_mediate()`, `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`, `pe_mediators()`, `pe_selection()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
# Contrasting mediation: active mediators whose effects cancel, so the
# total indirect effect is zero. Compare the two p-values: the benchmark
# Wald test's (pval_hdmm) is 0.19 while the PE test's (pval_pe) is tiny.
d <- simulate_mediation_data(n = 200, p = 60, pattern = "contrasting",
                             outcome = "continuous", c1 = 1)
pe_mediation(d$X, d$Y, d$M, outcome = "continuous")
# HBIC chose 0.05, the smallest value of the default grid, so the fit
# warned (class poemed_grid_boundary). A grid that extends below it and
# is spaced more finely moves the choice inside the grid, and the screen
# identifies the same three mediators.
finer <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous",
                      lambda_grid = seq(0.01, 10, length.out = 200))
attr(finer, "tuning")$lambda_selected
attr(finer, "active_mediators")
```

pe_mediation_linear *Power-Enhanced Mediation Test for a Continuous Outcome*

Description

Tests the global null hypothesis of no active mediator among a high-dimensional set of candidate mediators, for a continuous (linear-model) outcome, and reports both the benchmark Wald test on the total indirect effect and the power-enhanced (PE) test of Yu and Kelley (in press), which remains powerful when individual mediation effects are heterogeneous or contrasting. This is the continuous-outcome worker behind `pe_mediation()`; call it directly when the outcome is continuous.

Usage

```
pe_mediation_linear(
  X,
  Y,
  M,
  Z = NULL,
  method = c("Bonferroni", "BH", "BY"),
  scale = TRUE,
  error_level = 0.05,
  report_all_methods = FALSE,
  drop_constant = FALSE,
  lambda_grid = seq(0.05, 10, length.out = 100),
  lambda_grid_reduced = lambda_grid
)
```

Arguments

`X`	Numeric matrix of exposures with n rows and q columns (q is usually 1). A numeric vector is treated as a single exposure.
`Y`	Numeric outcome vector of length n. Continuous for `outcome = "continuous"`, coded 0/1 for `"binary"`, and nonnegative integer counts for `"count"`.
`M`	Numeric matrix of candidate mediators with n rows and p columns; p may exceed n. Column names, when present, label the mediators in the print footer and the mediator table; without them mediators are reported by column position.
`Z`	Optional numeric matrix of confounders with n rows. Default `NULL` (no confounders).
`method`	Multiplicity adjustment used to screen individual mediators in the PE component: `"Bonferroni"` (familywise error rate control, the default), `"BH"`, or `"BY"` (false discovery rate control by the procedures of Benjamini and Hochberg, 1995, and of Benjamini and Yekutieli, 2001; `"BH"` assumes independence or positive dependence among mediators, `"BY"` is valid under arbitrary dependence). See Details for the screening threshold each one uses.

`scale` Logical; if `TRUE` (default) the exposures, mediators, and confounders are each standardized to mean 0 and standard deviation 1 before fitting, as in the article, and a continuous outcome is centered but not rescaled. The total indirect effect is then the change in the outcome per standard deviation of each exposure: in the units of `Y` for a continuous outcome, in log odds for a binary outcome, and in log mean count for a count outcome. With `FALSE` the inputs are used exactly as supplied, and the total indirect effect is per unit of each exposure. The continuous-outcome fit has no intercept, and the default tuning grid was chosen for standardized columns, so `FALSE` is meant for inputs that are already standardized (and, for a continuous outcome, a `Y` that is already centered); on raw inputs it fits a different model and can even change the sign of the estimate. Because a continuous `Y` is not rescaled, the tuning grid is not expressed in the outcome's units, so the active mediators identified, both p-values, and the verdict depend on the units of `Y`: the same data with `Y` multiplied by 10 can identify a different set of mediators.

`error_level` Target error rate for the mediator-screening step, in $(0, 1)$. Interpreted as the familywise error rate when `method = "Bonferroni"` and as the false discovery rate when `method = "BH"` or `"BY"`. Default 0.05. This is distinct from the significance level used to test the global null (which is the user's choice when reading `pval_pe`). The PE test is asymptotically valid for any fixed value in $(0, 1)$, but in finite samples a larger value lets the screen pass more spurious mediators under the null and raises the PE test's size further above the benchmark's (see Details).

`report_all_methods` Logical; if `TRUE`, the identified active set is computed for all three multiplicity methods (Bonferroni, BH, BY) off the same fit and recorded for comparison, retrievable with `pe_selection()`. The `method` argument still drives the primary result. Default `FALSE`.

`drop_constant` Logical; how to handle constant (zero-variance) mediator columns, which cannot be standardized or carry signal. If `FALSE` (the default) they are an error; if `TRUE` they are dropped with a warning and the remaining mediators are renumbered back to their original column positions in the identified active set. Useful for small subgroups where some mediators happen to be constant.

`lambda_grid` Numeric vector of candidate tuning parameters for the SCAD penalty (Fan and Li, 2001) in the penalized mediator fit, which for a binary outcome is adaptively rescaled (see `pe_mediation_logistic()`). The fit is repeated at each value and the one minimizing the high-dimensional BIC (HBIC) of Wang, Kim, and Li (2013) is kept: whatever the grid, the value selected is the best of the values offered, since HBIC is minimized over the grid and not over every positive lambda. The default, `seq(0.05, 10, length.out = 100)` for every outcome family, is written out in the signature so that it can be edited in place. Every value is fit (the search is not adaptive), so the cost of a fit grows with the length of the grid. The grid can be tuned to the situation. Fewer values, or a range narrowed to where earlier fits put the HBIC minimum, make each fit faster, which matters when many fits are run (a simulation study, for example); a wider or finer grid is worth trying when the selected value is the smallest or largest value of the grid, in which case a non-empty fit warns (class `poemed_grid_boundary`) and the print footer, which reports the full model's choice, says so. A continuous outcome

also fits a reduced model, the outcome on the mediators and confounders without the exposure, tuned on the same grid by default (see `lambda_grid_reduced` in `pe_mediation_linear()`); its choice is `lambda_selected_reduced` in the `"tuning"` attribute, and the warning covers it as well, naming whether the full model, the reduced model, or both sat at the end. For a continuous outcome the grid is fixed while `Y` is centered but not rescaled, so the value HBIC chooses, and with it the active mediators identified, depends on the units of `Y` (see `scale`). The grid searched and the value chosen are reported in the `"tuning"` attribute and the print footer.

`lambda_grid_reduced`
: Numeric vector of candidate tuning parameters for the penalized fit of the reduced model, the outcome on the mediators and the confounders without the exposure, which the benchmark test's inference routine takes alongside the full-model fit. Searched and validated exactly as `lambda_grid`, and equal to it by default, matching the reference implementation's default `lamb_grid0 = lamb_grid`. Pass a different grid to tune the reduced fit on its own, which is worth doing when its HBIC choice sits at an end of the shared grid while the full model's does not. The value chosen is reported as `lambda_selected_reduced` in the `"tuning"` attribute. The reduced fit is specific to the continuous outcome, so this argument belongs to this function alone and `pe_mediation()` does not pass it through.

Details

The model is the linear mediation pair

$$Y = \alpha_m' M + \alpha_x' X + \alpha_z' Z + \varepsilon_y, \qquad M = \Gamma_x' X + \Gamma_z' Z + \varepsilon_m,$$

with total indirect effect $\beta = \Gamma_x \alpha_m$. Following Guo et al. (2023), the mediator coefficients α_m are estimated by partial penalized least squares with a SCAD penalty (Fan and Li, 2001), leaving the exposures and confounders unpenalized. The penalized fit is computed by the one-step local linear approximation of Zou and Li (2008) started from a lasso fit, as in Fan, Xue, and Zou (2014): a lasso fit, then a weighted lasso fit whose weights are the SCAD derivatives at the lasso coefficients. The tuning parameter is chosen by the high-dimensional BIC of Wang, Kim, and Li (2013). The fit has no intercept, because with `scale = TRUE` the outcome is centered and the other inputs standardized first (see `scale` in `pe_mediation()`).

The benchmark statistic is the Wald statistic $S_n = n\hat{\beta}'\hat{\Sigma}_\beta^{-1}\hat{\beta}$ of Guo et al. (2022), which extends the partially penalized Wald test of Guo et al. (2023) to observed confounders. The power-enhanced statistic of Yu and Kelley (in press), built on the power enhancement principle of Fan, Liao, and Yao (2015), adds the component J_m that accumulates the marginal signal from each mediator the screen identifies as active (see `pe_mediation()` for the rationale, the formula, the screening thresholds, and the test's finite-sample size). Both are referred to a chi-square distribution with q (the number of exposures) degrees of freedom, their common limit under the global null.

Value

A tidy `data.frame` of class `poemed_tbl` with rows `stat_hdmm` and `pval_hdmm` (the benchmark Wald test), `stat_pe`, `j_pe`, and `pval_pe` (the power-enhanced test and its PE component), `total_indirect_effect` (the estimated total indirect effect; one row per exposure when `q > 1`, suffixed with the exposure's column name when `X` has column names and with `_1`, `_2`, ... otherwise), `n_active_mediators` (the

number of active mediators the screen identified), df (the chi-square degrees of freedom, equal to q), n_candidate_mediators, and n_observations. With scale = TRUE the total indirect effect is in the units of Y per standard deviation of the exposure; with scale = FALSE it is in the units of Y per unit of the exposure as supplied. No confidence interval is reported: the article reports none, and the Wald interval built on the penalized estimate under-covers in finite samples.

Attributes: "active_mediators" (the column positions in M of the active mediators the screen identified; the print footer shows their column names when M has them), "mediator_names" and "exposure_names" (the column names of M and X, or NULL when unnamed), "mediator_table" (the per-mediator screening statistics for the penalized-selected mediators, see pe_mediators(); its t_exposure is the signed exposure-on-mediator statistic when q = 1 and the largest absolute one across exposures when q > 1, and its screen_p is the smallest screening p-value across exposures), "method", "error_level", "outcome", "p_terms" (the names of the rows that hold p-values, which the print method formats as p-values), "tuning" (a list with lambda_grid, the grid searched for the full model, the HBIC-selected lambda_selected, and at_lower_end and at_upper_end, TRUE when that selection sat at the smallest or largest value of lambda_grid; for a continuous outcome also lambda_grid_reduced and lambda_selected_reduced, the grid searched and the choice for the reduced model of the benchmark Wald test), "empty_fit", and "pe_by_method" (a data frame with one row per multiplicity method screened, giving that method's stat_pe, j_pe, pval_pe, and n_active, the number of active mediators it identified; it has one row unless report_all_methods = TRUE). With report_all_methods = TRUE there is also "selection_by_method", the identified active set under each method (see pe_selection()).

When the penalized fit that HBIC selects contains no mediator there is nothing to test: the function warns (a condition of class poemed_empty_fit), sets "empty_fit" to TRUE, and returns the same rows with both statistics 0, both p-values 1 (POEMED's reporting convention, under which an empty selection is a non-rejection; the article states only that $J_m = 0$ when the selection is empty), and a total indirect effect of 0. A p-value of 1 from such a fit is not evidence for the null. An empty fit is expected when no mediator is active. A non-empty fit warns (class poemed_grid_boundary) when its HBIC-selected lambda is the smallest or largest value of the grid searched, for the full model or, for a continuous outcome, for the reduced model of the benchmark Wald test, and the warning names the fit that sat there: the criterion is minimized over the grid alone, so extend the grid past that end and space its values more finely. The reduced model's choice is tested against the ends of lambda_grid_reduced, and the warning names that argument when it is not the full model's grid.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. citation("POEMED") gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Guo, X., Li, R., Liu, J., & Zeng, M. (2022). High-dimensional mediation analysis for selecting DNA methylation loci mediating childhood trauma and cortisol stress reactivity. *Journal of the American Statistical Association, 117*(539), 1110–1121. doi:10.1080/01621459.2022.2053136

Guo, X., Li, R., Liu, J., & Zeng, M. (2023). Statistical inference for linear mediation models with high-dimensional mediators and application to studying stock reaction to COVID-19 pandemic. *Journal of Econometrics, 235*(1), 166–179. doi:10.1016/j.jeconom.2022.03.001

Fan, J., Liao, Y., & Yao, J. (2015). Power enhancement in high-dimensional cross-sectional tests. *Econometrica, 83*(4), 1497–1541. doi:10.3982/ECTA12749

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association, 96*(456), 1348–1360. doi:10.1198/016214501753382273

Zou, H., & Li, R. (2008). One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics, 36*(4), 1509–1533. doi:10.1214/009053607000000802

Fan, J., Xue, L., & Zou, H. (2014). Strong oracle optimality of folded concave penalized estimation. *The Annals of Statistics, 42*(3), 819–849. doi:10.1214/13AOS1198

Wang, L., Kim, Y., & Li, R. (2013). Calibrating nonconvex penalized regression in ultra-high dimension. *The Annals of Statistics, 41*(5), 2505–2536. doi:10.1214/13AOS1159

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B, 57*(1), 289–300. doi:10.1111/j.25176161.1995.tb02031.x

Benjamini, Y., & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics, 29*(4), 1165–1188. doi:10.1214/aos/1013699998

See Also

`pe_mediation()` for the front end, `pe_mediation_logistic()` and `pe_mediation_poisson()` for binary and count outcomes, and `simulate_mediation_data()` to generate data for trying the method.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`, `pe_mediators()`, `pe_selection()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
# A contrasting setting: four active mediators whose indirect effects
# cancel, so the total indirect effect is zero. The Wald test does not
# reject here; the PE test rejects, and the screen identifies three of the
# four truly active mediators.
d <- simulate_mediation_data(n = 200, p = 60, pattern = "contrasting",
                             outcome = "continuous", c1 = 1)
pe_mediation_linear(d$X, d$Y, d$M)
# HBIC chose 0.05, the smallest value of the default grid, so the fit
# warned (class poemed_grid_boundary). A grid that extends below it and
# is spaced more finely moves the choice inside the grid, and the screen
# identifies the same three mediators.
finer <- pe_mediation_linear(d$X, d$Y, d$M,
                             lambda_grid = seq(0.01, 10, length.out = 200))
```

```
attr(finer, "tuning")$lambda_selected
attr(finer, "active_mediators")
```

`pe_mediation_logistic` *Power-Enhanced Mediation Test for a Binary Outcome*

Description

Tests the global null hypothesis of no active mediator for a binary (0/1) outcome modeled with logistic regression, reporting both the benchmark Wald test on the total indirect effect and the power-enhanced (PE) test of Yu and Kelley (in press). This is the binary-outcome worker behind `pe_mediation()`.

Usage

```
pe_mediation_logistic(
  X,
  Y,
  M,
  Z = NULL,
  method = c("Bonferroni", "BH", "BY"),
  scale = TRUE,
  error_level = 0.05,
  report_all_methods = FALSE,
  drop_constant = FALSE,
  lambda_grid = seq(0.05, 10, length.out = 100)
)
```

Arguments

`X`	Numeric matrix of exposures with n rows and q columns (q is usually 1). A numeric vector is treated as a single exposure.
`Y`	Numeric outcome vector of length n. Continuous for `outcome = "continuous"`, coded 0/1 for `"binary"`, and nonnegative integer counts for `"count"`.
`M`	Numeric matrix of candidate mediators with n rows and p columns; p may exceed n. Column names, when present, label the mediators in the print footer and the mediator table; without them mediators are reported by column position.
`Z`	Optional numeric matrix of confounders with n rows. Default `NULL` (no confounders).
`method`	Multiplicity adjustment used to screen individual mediators in the PE component: `"Bonferroni"` (familywise error rate control, the default), `"BH"`, or `"BY"` (false discovery rate control by the procedures of Benjamini and Hochberg, 1995, and of Benjamini and Yekutieli, 2001; `"BH"` assumes independence or positive dependence among mediators, `"BY"` is valid under arbitrary dependence). See Details for the screening threshold each one uses.

`scale` Logical; if `TRUE` (default) the exposures, mediators, and confounders are each standardized to mean 0 and standard deviation 1 before fitting, as in the article, and a continuous outcome is centered but not rescaled. The total indirect effect is then the change in the outcome per standard deviation of each exposure: in the units of `Y` for a continuous outcome, in log odds for a binary outcome, and in log mean count for a count outcome. With `FALSE` the inputs are used exactly as supplied, and the total indirect effect is per unit of each exposure. The continuous-outcome fit has no intercept, and the default tuning grid was chosen for standardized columns, so `FALSE` is meant for inputs that are already standardized (and, for a continuous outcome, a `Y` that is already centered); on raw inputs it fits a different model and can even change the sign of the estimate. Because a continuous `Y` is not rescaled, the tuning grid is not expressed in the outcome's units, so the active mediators identified, both p-values, and the verdict depend on the units of `Y`: the same data with `Y` multiplied by 10 can identify a different set of mediators.

`error_level` Target error rate for the mediator-screening step, in $(0, 1)$. Interpreted as the familywise error rate when `method = "Bonferroni"` and as the false discovery rate when `method = "BH"` or `"BY"`. Default 0.05. This is distinct from the significance level used to test the global null (which is the user's choice when reading `pval_pe`). The PE test is asymptotically valid for any fixed value in $(0, 1)$, but in finite samples a larger value lets the screen pass more spurious mediators under the null and raises the PE test's size further above the benchmark's (see Details).

`report_all_methods` Logical; if `TRUE`, the identified active set is computed for all three multiplicity methods (Bonferroni, BH, BY) off the same fit and recorded for comparison, retrievable with `pe_selection()`. The `method` argument still drives the primary result. Default `FALSE`.

`drop_constant` Logical; how to handle constant (zero-variance) mediator columns, which cannot be standardized or carry signal. If `FALSE` (the default) they are an error; if `TRUE` they are dropped with a warning and the remaining mediators are renumbered back to their original column positions in the identified active set. Useful for small subgroups where some mediators happen to be constant.

`lambda_grid` Numeric vector of candidate tuning parameters for the SCAD penalty (Fan and Li, 2001) in the penalized mediator fit, which for a binary outcome is adaptively rescaled (see `pe_mediation_logistic()`). The fit is repeated at each value and the one minimizing the high-dimensional BIC (HBIC) of Wang, Kim, and Li (2013) is kept: whatever the grid, the value selected is the best of the values offered, since HBIC is minimized over the grid and not over every positive lambda. The default, `seq(0.05, 10, length.out = 100)` for every outcome family, is written out in the signature so that it can be edited in place. Every value is fit (the search is not adaptive), so the cost of a fit grows with the length of the grid. The grid can be tuned to the situation. Fewer values, or a range narrowed to where earlier fits put the HBIC minimum, make each fit faster, which matters when many fits are run (a simulation study, for example); a wider or finer grid is worth trying when the selected value is the smallest or largest value of the grid, in which case a non-empty fit warns (class `poemed_grid_boundary`) and the print footer, which reports the full model's choice, says so. A continuous outcome

also fits a reduced model, the outcome on the mediators and confounders without the exposure, tuned on the same grid by default (see `lambda_grid_reduced` in `pe_mediation_linear()`); its choice is `lambda_selected_reduced` in the `"tuning"` attribute, and the warning covers it as well, naming whether the full model, the reduced model, or both sat at the end. For a continuous outcome the grid is fixed while `Y` is centered but not rescaled, so the value HBIC chooses, and with it the active mediators identified, depends on the units of `Y` (see `scale`). The grid searched and the value chosen are reported in the `"tuning"` attribute and the print footer.

Details

The outcome follows a logistic mediation model, $\operatorname{logit} P(Y = 1 \mid M, X, Z) = \alpha_m' M + \alpha_x' X + \alpha_z' Z$, with a linear mediator model as in `pe_mediation_linear()`. Following Guo et al. (2024), the mediator coefficients are estimated by partial penalized likelihood, leaving the exposures and confounders unpenalized, with the tuning parameter chosen by the high-dimensional BIC of Wang, Kim, and Li (2013). The benchmark Wald test on the total indirect effect $\beta = \Gamma_x \alpha_m$ is also that of Guo et al. (2024). The power-enhanced test of Yu and Kelley (in press), built on the power enhancement principle of Fan, Liao, and Yao (2015), adds the component J_m, formed on the logit (link) scale exactly as in the continuous case (see `pe_mediation()`). The total indirect effect here is on the log-odds scale, not the probability scale: with `scale = TRUE` it is the change in the log odds per standard deviation of the exposure.

The penalty is SCAD (Fan and Li, 2001) with an adaptive rescaling, not the plain SCAD penalty. The fit is an iteratively reweighted coordinate descent. Each coordinate update applies the SCAD thresholds to the mediator's weighted working score and only then divides by its curvature v_j (the mean of the squared standardized mediator weighted by the working weights, at most 1/4 for a logistic fit). The fit is therefore a stationary point of the penalized likelihood with penalty $\sum_j p_\lambda(v_j|\alpha_{m,j}|)/v_j$, not with the plain SCAD penalty $\sum_j p_\lambda(|\alpha_{m,j}|)$. The condition for a coefficient to be zero is the same as under plain SCAD (its score is at most λ in absolute value), but a nonzero coefficient escapes shrinkage only beyond $\gamma\lambda/v_j$ rather than $\gamma\lambda$ ($\gamma = 3.7$), so coefficients of moderate size are shrunk more than plain SCAD would shrink them.

Value

A tidy `data.frame` of class `poemed_tbl` with the same rows and attributes as `pe_mediation_linear()`, except that the `"tuning"` list has no reduced-model entries (`lambda_grid_reduced` and `lambda_selected_reduced`): the binary fit has no reduced model.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Guo, X., Li, R., Liu, J., & Zeng, M. (2024). Estimations and tests for generalized mediation models with high-dimensional potential mediators. *Journal of Business & Economic Statistics, 42*(1), 243–256. doi:10.1080/07350015.2023.2174548

Fan, J., Liao, Y., & Yao, J. (2015). Power enhancement in high-dimensional cross-sectional tests. *Econometrica, 83*(4), 1497–1541. doi:10.3982/ECTA12749

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association, 96*(456), 1348–1360. doi:10.1198/016214501753382273

Wang, L., Kim, Y., & Li, R. (2013). Calibrating nonconvex penalized regression in ultra-high dimension. *The Annals of Statistics, 41*(5), 2505–2536. doi:10.1214/13AOS1159

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B, 57*(1), 289–300. doi:10.1111/j.25176161.1995.tb02031.x

Benjamini, Y., & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics, 29*(4), 1165–1188. doi:10.1214/aos/1013699998

See Also

`pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_poisson()`.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_poisson()`, `pe_mediators()`, `pe_selection()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
# Contrasting mediation with a binary outcome: the benchmark Wald test
# does not reject, the PE test does, and the screen identifies one of the
# two truly active mediators.
d <- simulate_mediation_data(n = 200, p = 60, pattern = "contrasting",
                             outcome = "binary", c1 = 1, c2 = 1)
pe_mediation_logistic(d$X, d$Y, d$M)
# HBIC chose 0.05, the smallest value of the default grid, so the fit
# warned (class poemed_grid_boundary); on a grid that extends below it
# and is spaced more finely, seq(0.01, 10, length.out = 200), the screen
# identifies the same mediator.
```

`pe_mediation_poisson` *Power-Enhanced Mediation Test for a Count Outcome*

Description

Tests the global null hypothesis of no active mediator for a count outcome modeled with Poisson (log-link) regression, reporting both the benchmark Wald test on the total indirect effect and the power-enhanced (PE) test of Yu and Kelley (in press). This is the count-outcome worker behind `pe_mediation()`.

Usage

```
pe_mediation_poisson(
  X,
  Y,
  M,
  Z = NULL,
  method = c("Bonferroni", "BH", "BY"),
  scale = TRUE,
  error_level = 0.05,
  report_all_methods = FALSE,
  drop_constant = FALSE,
  lambda_grid = seq(0.05, 10, length.out = 100)
)
```

Arguments

`X`	Numeric matrix of exposures with n rows and q columns (q is usually 1). A numeric vector is treated as a single exposure.
`Y`	Numeric outcome vector of length n. Continuous for `outcome = "continuous"`, coded 0/1 for `"binary"`, and nonnegative integer counts for `"count"`.
`M`	Numeric matrix of candidate mediators with n rows and p columns; p may exceed n. Column names, when present, label the mediators in the print footer and the mediator table; without them mediators are reported by column position.
`Z`	Optional numeric matrix of confounders with n rows. Default `NULL` (no confounders).
`method`	Multiplicity adjustment used to screen individual mediators in the PE component: `"Bonferroni"` (familywise error rate control, the default), `"BH"`, or `"BY"` (false discovery rate control by the procedures of Benjamini and Hochberg, 1995, and of Benjamini and Yekutieli, 2001; `"BH"` assumes independence or positive dependence among mediators, `"BY"` is valid under arbitrary dependence). See Details for the screening threshold each one uses.
`scale`	Logical; if `TRUE` (default) the exposures, mediators, and confounders are each standardized to mean 0 and standard deviation 1 before fitting, as in the article, and a continuous outcome is centered but not rescaled. The total indirect effect is then the change in the outcome per standard deviation of each exposure: in the units of `Y` for a continuous outcome, in log odds for a binary outcome, and in log mean count for a count outcome. With `FALSE` the inputs are used exactly as supplied, and the total indirect effect is per unit of each exposure. The continuous-outcome fit has no intercept, and the default tuning grid was chosen

for standardized columns, so FALSE is meant for inputs that are already standardized (and, for a continuous outcome, a Y that is already centered); on raw inputs it fits a different model and can even change the sign of the estimate. Because a continuous Y is not rescaled, the tuning grid is not expressed in the outcome's units, so the active mediators identified, both p-values, and the verdict depend on the units of Y: the same data with Y multiplied by 10 can identify a different set of mediators.

error_level
: Target error rate for the mediator-screening step, in $(0, 1)$. Interpreted as the familywise error rate when method = "Bonferroni" and as the false discovery rate when method = "BH" or "BY". Default 0.05. This is distinct from the significance level used to test the global null (which is the user's choice when reading pval_pe). The PE test is asymptotically valid for any fixed value in $(0, 1)$, but in finite samples a larger value lets the screen pass more spurious mediators under the null and raises the PE test's size further above the benchmark's (see Details).

report_all_methods
: Logical; if TRUE, the identified active set is computed for all three multiplicity methods (Bonferroni, BH, BY) off the same fit and recorded for comparison, retrievable with pe_selection(). The method argument still drives the primary result. Default FALSE.

drop_constant
: Logical; how to handle constant (zero-variance) mediator columns, which cannot be standardized or carry signal. If FALSE (the default) they are an error; if TRUE they are dropped with a warning and the remaining mediators are renumbered back to their original column positions in the identified active set. Useful for small subgroups where some mediators happen to be constant.

lambda_grid
: Numeric vector of candidate tuning parameters for the SCAD penalty (Fan and Li, 2001) in the penalized mediator fit, which for a binary outcome is adaptively rescaled (see pe_mediation_logistic()). The fit is repeated at each value and the one minimizing the high-dimensional BIC (HBIC) of Wang, Kim, and Li (2013) is kept: whatever the grid, the value selected is the best of the values offered, since HBIC is minimized over the grid and not over every positive lambda. The default, seq(0.05, 10, length.out = 100) for every outcome family, is written out in the signature so that it can be edited in place. Every value is fit (the search is not adaptive), so the cost of a fit grows with the length of the grid. The grid can be tuned to the situation. Fewer values, or a range narrowed to where earlier fits put the HBIC minimum, make each fit faster, which matters when many fits are run (a simulation study, for example); a wider or finer grid is worth trying when the selected value is the smallest or largest value of the grid, in which case a non-empty fit warns (class poemed_grid_boundary) and the print footer, which reports the full model's choice, says so. A continuous outcome also fits a reduced model, the outcome on the mediators and confounders without the exposure, tuned on the same grid by default (see lambda_grid_reduced in pe_mediation_linear()); its choice is lambda_selected_reduced in the "tuning" attribute, and the warning covers it as well, naming whether the full model, the reduced model, or both sat at the end. For a continuous outcome the grid is fixed while Y is centered but not rescaled, so the value HBIC chooses, and with it the active mediators identified, depends on the units of Y (see scale). The grid searched and the value chosen are reported in the "tuning" attribute and the print footer.

Details

The outcome follows a Poisson mediation model, $Y \mid M, X, Z \sim \mathrm{Poisson}(\exp(\alpha_m' M + \alpha_x' X + \alpha_z' Z))$, with a linear mediator model as in `pe_mediation_linear()`. Estimation, tuning, and the benchmark Wald test follow the generalized-outcome framework of Guo et al. (2024): the mediator coefficients are estimated by partial penalized likelihood with a SCAD penalty (Fan and Li, 2001), leaving the exposures and confounders unpenalized, with the tuning parameter chosen by the high-dimensional BIC of Wang, Kim, and Li (2013). The penalized Poisson fit is a glmnet lasso fit followed by one weighted lasso step on a quadratic approximation of the Poisson likelihood at the lasso estimate, with the SCAD derivatives at the lasso coefficients as weights: the one-step local linear approximation of Zou and Li (2008), started from the lasso as in Fan, Xue, and Zou (2014). The power-enhanced test of Yu and Kelley (in press), built on the power enhancement principle of Fan, Liao, and Yao (2015), adds the component J_m (see `pe_mediation()`). The total indirect effect $\beta = \Gamma_x \alpha_m$ is on the log-mean (link) scale, not the count scale: with `scale = TRUE` it is the change in the log mean count per standard deviation of the exposure.

Value

A tidy `data.frame` of class `poemed_tbl` with the same rows and attributes as `pe_mediation_linear()`, except that the `"tuning"` list has no reduced-model entries (`lambda_grid_reduced` and `lambda_selected_reduced`): the count fit has no reduced model.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Guo, X., Li, R., Liu, J., & Zeng, M. (2024). Estimations and tests for generalized mediation models with high-dimensional potential mediators. *Journal of Business & Economic Statistics, 42*(1), 243–256. doi:10.1080/07350015.2023.2174548

Fan, J., Liao, Y., & Yao, J. (2015). Power enhancement in high-dimensional cross-sectional tests. *Econometrica, 83*(4), 1497–1541. doi:10.3982/ECTA12749

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association, 96*(456), 1348–1360. doi:10.1198/016214501753382273

Zou, H., & Li, R. (2008). One-step sparse estimates in nonconcave penalized likelihood models. *The Annals of Statistics, 36*(4), 1509–1533. doi:10.1214/009053607000000802

Fan, J., Xue, L., & Zou, H. (2014). Strong oracle optimality of folded concave penalized estimation. *The Annals of Statistics, 42*(3), 819–849. doi:10.1214/13AOS1198

Wang, L., Kim, Y., & Li, R. (2013). Calibrating nonconvex penalized regression in ultra-high dimension. *The Annals of Statistics, 41*(5), 2505–2536. doi:10.1214/13AOS1159

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society, Series B, 57*(1), 289–300. doi:10.1111/j.25176161.1995.tb02031.x

Benjamini, Y., & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics, 29*(4), 1165–1188. doi:10.1214/aos/1013699998

See Also

`pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_logistic()`.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediators()`, `pe_selection()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
# Contrasting mediation with a count outcome: the benchmark Wald test
# does not reject, the PE test does, and the screen identifies both truly
# active mediators (4 and 5).
d <- simulate_mediation_data(n = 200, p = 60, pattern = "contrasting",
                             outcome = "count", c1 = 0.5, c2 = 0.4)
pe_mediation_poisson(d$X, d$Y, d$M)
```

`pe_mediators` *Per-Mediator Detail Behind a Power-Enhanced Mediation Test*

Description

Returns the per-mediator table that explains a `pe_mediation()` result. The table has one row for each candidate mediator the penalized fit selected (retained with a nonzero coefficient). Each row gives the mediator's two path statistics, its screening p-value, and whether the screen identified it as active. This is the "why" behind the active mediators identified, so a user can see which of those candidates were close to the screening threshold and which were screened out.

Usage

```
pe_mediators(fit)
```

Arguments

`fit`	A result from `pe_mediation()` or one of its workers.

Details

Two steps narrow the mediators, and the table keeps them apart. The penalized fit selects some of the candidate mediators, which are the rows of the table. The multiplicity screen then identifies active mediators among those candidates, which are the rows whose `active` column is `TRUE`. A row with `active = FALSE` is therefore a candidate that the screen did not identify as active, not a mediator the penalized fit dropped.

Value

A tidy `data.frame` (class `poemed_tbl`) with columns:

mediator Column position of the mediator in the original `M`.

name The mediator's column name in `M`; present only when `M` has column names.

t_outcome The standardized mediator-on-outcome statistic (the $M \to Y$ path, $\hat{\alpha}_{m,j}/\hat{\sigma}_{m,j}$).

t_exposure The standardized exposure-on-mediator statistic (the $X \to M$ path). With one exposure it is the signed *t* statistic, so a negative value marks a negative path. With several exposures it is the largest absolute *t* statistic across the exposures, so it is never negative and does not show the path's direction; regress the mediator on the exposures and any confounders (for example with `lm()`) to see the signs.

screen_p The screening p-value, the larger of the two path p-values (the smaller of these across exposures when there are several). A mediator is identified as active when this clears the multiplicity threshold.

active Logical; `TRUE` when the screen identified the mediator as active under the fit's `method`, so the `TRUE` rows are the fit's active mediators (its `"active_mediators"` attribute). Every row is a candidate the penalized fit selected, so `FALSE` marks a candidate the screen did not identify as active.

The table is empty when the penalized fit selected no candidate mediators.

Author(s)

Xiufan Yu and Ken Kelley

See Also

`pe_mediation()`, `pe_selection()`.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`, `pe_selection()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 200, p = 60, outcome = "continuous",
                             pattern = "contrasting", c1 = 1)
fit <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous")
# HBIC chose the smallest value of the default grid here, so the fit
# warned; see the lambda_grid argument of ?pe_mediation.
pe_mediators(fit)
```

pe_power_curve *Monte Carlo Size and Power Curve for the PE Mediation Tests*

Description

Runs the article's simulation design: for each value of the signal-strength scale c_1 on a grid, it simulates many data sets, applies both the benchmark Wald test and the power-enhanced test, and returns their empirical rejection rates. At $c_1 = 0$ (the global null of no mediation) the rejection rate estimates the Type I error rate; at nonzero c_1 it estimates power. The headline finding is visible directly in the table: under a contrasting pattern the power-enhanced test climbs toward one as $|c_1|$ grows, while the benchmark test, whose target is a total indirect effect that cancels, gains power much more slowly.

Usage

```
pe_power_curve(
  n,
  p,
  outcome = c("continuous", "binary", "count"),
  pattern = c("homogeneous", "contrasting", "heterogeneous"),
  c1_grid = seq(0, 1, by = 0.25),
  c2 = 0.5,
  n_rep = 100,
  alpha_level = 0.05,
  method = c("Bonferroni", "BH", "BY"),
  error_level = 0.05,
  lambda_grid = seq(0.05, 10, length.out = 100),
  outcome_args = list(),
  cores = 1L,
  seed = NULL,
  progress = FALSE
)
```

Arguments

n, p	Number of observations and candidate mediators per data set.
outcome	Outcome type passed to `simulate_mediation_data()`: `"continuous"`, `"binary"`, or `"count"`.
pattern	Mediation pattern: `"homogeneous"` or `"contrasting"` (`"heterogeneous"` is a synonym for `"contrasting"`).
c1_grid	Numeric vector of signal-strength scales to sweep. Default `seq(0, 1, by = 0.25)`. Include 0 to estimate the Type I error rate; the article also uses negative values.
c2	Direct effect used in the simulation. Default 0.5, the value of the article's linear simulations; its logistic figures use 1 and its Poisson figures 0.4.

`n_rep` Number of Monte Carlo replications per grid point. Default 100. The article uses 1000.

`alpha_level` Significance level for the global test. A replication counts as a rejection when its p-value is at most `alpha_level`. Default 0.05.

`method`, `error_level`
Passed to `pe_mediation()`: the multiplicity method and target error rate for the mediator-screening step.

`lambda_grid` Passed to `pe_mediation()`: the tuning grid for the penalized fit, by default `seq(0.05, 10, length.out = 100)` (see Details). A shorter or narrower grid speeds a study up.

`outcome_args` A named list of further arguments forwarded to `simulate_mediation_data()` (for example `rho`, `q`, `d`, `tau`, `alpha_m`). Default empty. With `d` above 0 and no `alpha_z` or `Gamma_z`, the simulator draws the confounder coefficients afresh in every replication. Each name must identify one argument of `simulate_mediation_data()`; an unknown or ambiguous name stops with an error rather than reaching the simulator. The design arguments this function sets itself (`n`, `p`, `outcome`, `pattern`, `c1`, `c2`) and `seed` may not appear here, whether spelled in full or abbreviated; a `seed` inside `outcome_args` would make every replication draw the same data.

`cores` Number of CPU cores for the Monte Carlo replications. Values above 1 fork via the base **parallel** package (Unix only). With parallel cores the replications run on the `"L'Ecuyer-CMRG"` generator (the caller's generator kind is restored on exit), so a seeded parallel run is reproducible across runs at the same `cores` but need not match a serial run. The parallel streams are handed out afresh from the same generator state at every point of `c1_grid`, so with `cores` above 1 each grid point replays the same `n_rep` random draws (common random numbers): the rates at neighboring grid points share their Monte Carlo noise, and the curve looks smoother than the separate errors of its points suggest. With `cores = 1` every grid point draws fresh data. Default 1.

`seed` Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. `NULL` (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

`progress` `TRUE` or `FALSE`; if `TRUE`, print a line per grid point as it completes. Default `FALSE`.

Details

A rejection rate from `n_rep` replications carries a Monte Carlo standard error of about $\sqrt{r(1-r)/n_{rep}}$; at 100 replications a rate near 0.05 is known to about 0.02 and a rate near 0.5 to about 0.05. The article uses 1000 replications. A replication whose penalized fit selects no mediator counts as a non-rejection (both p-values are 1), POEMED's reporting convention (the article states only that $J_m = 0$ when the selection is empty); the number of such replications is reported in `n_empty`. A replication whose fit fails outright is dropped from the denominator and reported in `n_valid`, with a warning that quotes the first error. A design that cannot be simulated (for example a `p` too small for the pattern, or a misspelled name in `outcome_args`) stops before any replication runs, with the same message at any value of `cores`.

The tuning grid matters for what these curves show. Every replication searches the same grid: the package default `seq(0.05, 10, length.out = 100)`, or the grid passed as `lambda_grid` (for a

continuous outcome the full model and the reduced model both search it; the study functions do not take a separate `lambda_grid_reduced`). A study of many replications is where a grid tuned to the situation pays: fewer values, or a range narrowed to where the HBIC minimum has been seen to fall, make each fit faster at the price of a coarser choice, and whatever the grid, each fit keeps the best value the grid offers. The single-fit warning for a choice at an end of the grid (class `poemed_grid_boundary`) is not raised inside a study, which searches a fixed grid by design.

Under contrasting mediation the benchmark can have power although the total indirect effect is zero: a penalized fit that keeps only part of the canceling set leaves a total indirect effect, over the kept mediators, that no longer cancels, and how often that happens depends on the tuning grid. The article's logistic and Poisson figures fix c_2 at 1 and 0.4, not at this function's default of 0.5, so pass `c2` to match them.

Value

A tidy `data.frame` of class `poemed_tbl` with one row per grid point and columns `c1`, `rejection_hdmm`, and `rejection_pe` (the empirical rejection rates of the benchmark and power-enhanced tests), `n_valid` (replications whose fit succeeded, the denominator of the rates), and `n_empty` (replications whose penalized fit selected no mediator, counted as non-rejections). The simulation settings are recorded in the attributes `outcome`, `pattern`, `n`, `p`, `n_rep`, `alpha_level`, `error_level`, and `lambda_grid`.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

`pe_mediation()`, `simulate_mediation_data()`.

Other mediation simulation: `guo_calibration`, `mediation_ar1_cov()`, `pe_identification_study()`, `pe_simulation_study()`, `plot.poemed_tbl()`, `simulate_guo_mediation()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
# n_rep = 5 keeps this example fast; a rate from five replications has a
# Monte Carlo standard error of up to 0.22, so read the shape, not the
# numbers. A reported study uses the article's design (n = 300, p = 500)
# and 1000 replications. A study is also where a shorter tuning grid
# pays: every replication searches the whole grid, so this one passes 20
```

```
# values from 0.05 to 2 instead of the default 100 values from 0.05 to 10
# (see the lambda_grid argument of ?pe_mediation).
set.seed(113)
pe_power_curve(n = 200, p = 60, outcome = "continuous",
               pattern = "contrasting", c1_grid = c(0, 0.5, 1),
               n_rep = 5, lambda_grid = seq(0.05, 2, length.out = 20))
```

`pe_selection` *Active Mediators Identified Under Each Multiplicity Method*

Description

Summarizes which mediators a `pe_mediation()` fit identified as active. When the fit was produced with `report_all_methods = TRUE`, this reports the identified active set under each of the three multiplicity methods (Bonferroni for familywise error rate control, Benjamini-Hochberg and Benjamini-Yekutieli for false discovery rate control), so their conservativeness can be compared on the same fit. Otherwise it reports the single method that was used.

Usage

```
pe_selection(fit)
```

Arguments

`fit`	A result from `pe_mediation()` or one of its workers.

Value

A tidy `data.frame` of class `poemed_tbl` (so its p-values print to fixed decimals) with one row per multiplicity method and columns `method`, `n_active` (number of active mediators identified), `j_pe` (the power enhancement component J_m under that method's screen), `stat_pe` and `pval_pe` (the resulting power-enhanced statistic M_{PE} and its p-value), and `active_mediators` (their column names when `M` has them, otherwise their column positions, comma-separated, or `"none"`). Because each multiplicity method screens a different identified set into J_m, the `pval_pe` column gives the PE / PE_BH / PE_BY global p-values side by side, as the article's extended data-analysis tables report them.

Author(s)

Xiufan Yu and Ken Kelley

See Also

`pe_mediation()` (its `report_all_methods` argument), `pe_mediators()`.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`, `pe_mediators()`, `summary.poemed_tbl()`

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 300, p = 100, outcome = "continuous",
                             pattern = "homogeneous", c1 = 1)
fit <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous",
                    report_all_methods = TRUE)
# On this homogeneous design (truly active mediators 1 to 5) the screens
# differ: Bonferroni, which controls the familywise error rate,
# identifies three mediators as active (3, 4, and 5); BH, the false
# discovery rate screen for independent or positively dependent
# mediators, identifies four (2 to 5); BY, valid under arbitrary
# dependence, identifies the same three as Bonferroni.
pe_selection(fit)
d$active_mediators
# HBIC chose 0.05, the smallest value of the default grid, so the fit
# warned. On a grid that extends below it and is spaced more finely,
# seq(0.01, 10, length.out = 200), it chooses about 0.11, and the
# screens then identify 3 and 4 (Bonferroni) and 2, 3, and 4 (BH and BY):
# the identified active set depends on the grid as well as on the screen.
```

---

pe_simulation_study *Run the Power Study Under Several Mediation Patterns at Once*

Description

A convenience wrapper around `pe_power_curve()` that runs the size and power study under more than one mediation pattern in a single call and stacks the results, so a homogeneous and a contrasting curve (the two panels of a figure in the article) come back together. Each pattern is swept over the same signal grid.

Usage

```
pe_simulation_study(
  n,
  p,
  outcome = c("continuous", "binary", "count"),
  patterns = c("homogeneous", "contrasting"),
  c1_grid = seq(0, 1, by = 0.25),
  c2 = 0.5,
  n_rep = 100,
  alpha_level = 0.05,
  method = c("Bonferroni", "BH", "BY"),
  error_level = 0.05,
  lambda_grid = seq(0.05, 10, length.out = 100),
  cores = 1L,
  seed = NULL
)
```

Arguments

- `n`, `p` — Observations and candidate mediators per simulated data set.
- `outcome` — Outcome type: `"continuous"`, `"binary"`, or `"count"`.
- `patterns` — Character vector of mediation patterns to run, each named once. The default runs `"homogeneous"` and `"contrasting"`.
- `c1_grid`, `c2`, `n_rep`, `alpha_level`, `method`, `error_level`, `lambda_grid`, `cores` — Passed to `pe_power_curve()`.
- `seed` — Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. `NULL` (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

Details

Everything `pe_power_curve()` says about Monte Carlo error, empty fits, and the tuning grid applies here; the article's logistic and Poisson figures fix `c2` at 1 and 0.4. The patterns take their data from one continuing random stream, so no two patterns share their draws. With `cores` above 1 the points of `c1_grid` within a pattern do share them (see `cores` on `pe_power_curve()`).

Value

A tidy `data.frame` (class `poemed_tbl`) with a leading `pattern` column and the `pe_power_curve()` columns (`c1`, `rejection_hdmm`, `rejection_pe`, `n_valid`, `n_empty`) for each pattern. The settings are recorded in the attributes `outcome`, `n`, `p`, `n_rep`, `alpha_level`, `error_level`, and `lambda_grid`.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

See Also

`pe_power_curve()` for a single pattern, `plot.poemed_tbl()` to draw one pattern's curves.

Other mediation simulation: `guo_calibration`, `mediation_ar1_cov()`, `pe_identification_study()`, `pe_power_curve()`, `plot.poemed_tbl()`, `simulate_guo_mediation()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
# n_rep = 5 keeps this example fast; a rate from five replications has a
# Monte Carlo standard error of up to 0.22, so read the shape, not the
# numbers. A reported study uses the article's design (n = 300, p = 500)
# and 1000 replications. A study is also where a shorter tuning grid
```

```
# pays: every replication searches the whole grid, so this one passes 20
# values from 0.05 to 2 instead of the default 100 values from 0.05 to 10
# (see the lambda_grid argument of ?pe_mediation).
set.seed(113)
pe_simulation_study(n = 120, p = 50, outcome = "continuous",
                    c1_grid = c(0, 1), n_rep = 5,
                    lambda_grid = seq(0.05, 2, length.out = 20))
```

`plot.poemed_tbl` *Plot a POEMED Result*

Description

A base-graphics plot for the two kinds of table POEMED returns. For a power curve from `pe_power_curve()` it draws the empirical rejection rate of the benchmark and power-enhanced tests against the signal-strength grid, with the nominal level marked. For a single `pe_mediation()` fit it draws the per-mediator screening evidence ($-\log_{10}$ of the screening p-value) for each candidate mediator the penalized fit selected, with the ones the screen identified as active highlighted (black by default, "active" in the legend; the other candidates are gray, "screened out"), so it is clear which mediators drove the result.

Usage

```
## S3 method for class 'poemed_tbl'
plot(x, ...)
```

Arguments

`x`	A `poemed_tbl` from `pe_power_curve()` or `pe_mediation()`.
`...`	Further graphical parameters passed to the underlying `plot.default()` call. They override the defaults, which are `xlab`, `ylab`, `pch`, `col`, and `ylim` (from 0 to 1.3 times the tallest point, leaving room for the legend) in the per-mediator view and `xlab`, `ylab`, `pch`, `type`, and `ylim` in the power-curve view, so `plot(fit, xlab = "mediator", col = "red")` works. A supplied `pch`, `col`, or `lty` is carried into the legend. In the per-mediator view the x axis is labeled with the mediators' names or positions; passing `xaxt` replaces that axis with the one `xaxt` asks for.

Details

A screening p-value is stored as exactly 0 only when it underflows the smallest double, which takes both path statistics at about 38 or more in absolute value (the table prints it as `< 0.0001`, like any p-value below that floor). Its $-\log_{10}$ is infinite, so the per-mediator view draws such a point at a cap instead. The cap is the larger of $-\log_{10}$ of the machine epsilon (about 15.65) and the tallest finite point. A capped point is drawn as a triangle, a dotted line marks the cap, and the legend says so. The true height of a capped point is at least the cap. Only the drawing changes; the stored `screen_p` values (see `pe_mediators()`) are untouched.

Value

x, invisibly. Called for the plot it draws. A table that is neither a fit with a per-mediator table nor a power curve draws nothing and gives a message.

Author(s)

Xiufan Yu and Ken Kelley

See Also

pe_power_curve(), pe_mediation(), pe_mediators().

Other mediation simulation: guo_calibration, mediation_ar1_cov(), pe_identification_study(), pe_power_curve(), pe_simulation_study(), simulate_guo_mediation(), simulate_mediation_data(), ss_power_pe_mediation()

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 200, p = 60, outcome = "continuous",
                             pattern = "contrasting", c1 = 1)
fit <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous")
# HBIC chose the smallest value of the default grid here, so the fit
# warned; see the lambda_grid argument of ?pe_mediation.
plot(fit)
plot(fit, main = "Screening evidence", xlab = "candidate mediator", pch = 15)
```

---

poemed_broom *Broom Verbs for POEMED Results*

Description

tidy() and glance() methods so a POEMED result composes with the broom ecosystem; the generics are re-exported, so tidy(fit) and glance(fit) work after library(POEMED) alone. For a pe_mediation() fit, tidy() returns the per-mediator table (see pe_mediators()) and glance() returns a one-row summary of the two tests and the total indirect effect. For a pe_power_curve() table both return the table itself (it is already one row per grid point).

Usage

```
## S3 method for class 'poemed_tbl'
tidy(x, ...)

## S3 method for class 'poemed_tbl'
glance(x, ...)

tidy(x, ...)

glance(x, ...)
```

Arguments

x	A poemed_tbl result.
...	Unused, for generic compatibility.

Value

A data.frame. For a fit, tidy() has the columns of pe_mediators() and glance() the columns stat_hdmm, pval_hdmm, stat_pe, pval_pe, total_indirect_effect (one column per exposure when there are several, suffixed with the exposure's column name or, for an unnamed X, _1, _2, ...), n_active_mediators, and n_observations.

Author(s)

Xiufan Yu and Ken Kelley

See Also

pe_mediation(), pe_mediators().

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 120, p = 40, pattern = "contrasting",
                             outcome = "continuous", c1 = 1)
fit <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous")
# HBIC chose the smallest value of the default grid here, so the fit
# warned; see the lambda_grid argument of ?pe_mediation.
tidy(fit)
glance(fit)
```

---

poemed_tbl *Tidy Printing for POEMED Result Tables*

Description

Every estimation and testing function in **POEMED** returns a tidy data.frame with a term column and one or more numeric columns. A single numeric column routinely holds quantities on very different scales: a whole-number count of mediators next to a chi-square statistic in the hundreds next to a p-value of 10^{-12}. The base print.data.frame method formats a whole column with one common format, which forces either a wall of trailing zeros or a slide into scientific notation. The poemed_tbl class supplies print and format methods that format each value on its own terms: whole numbers (counts, sample sizes, mediator indices) print without a decimal part, other values print to a few significant figures, and p-values print to a fixed number of decimal places with a "< 0.0001" floor.

Usage

```
## S3 method for class 'poemed_tbl'
format(x, digits = getOption("poemed.digits", 4L), digits_p = 4L, ...)

## S3 method for class 'poemed_tbl'
print(x, digits = getOption("poemed.digits", 4L), digits_p = 4L, ...)

## S3 method for class 'poemed_tbl'
x[...]

## S3 method for class 'poemed_tbl'
rbind(..., deparse.level = 1)
```

Arguments

`x`	A `poemed_tbl` object (a tidy `data.frame` returned by a POEMED function).
`digits`	Number of significant figures for non-integer values, a single whole number from 1 to 22 (the range base R's `format()` accepts). Defaults to `getOption("poemed.digits", 4L)`, so `options(poemed.digits = 6)` changes it for the session.
`digits_p`	Number of decimal places for p-values, a single whole number from 1 to 15 (about the number of decimal digits a double holds reliably). Defaults to 4. A p-value below `10^(-digits_p)` prints as "< " followed by that bound, so "< 0.0001" at the default and "< 0.01" with `digits_p = 2`.
`...`	For `print`, further arguments passed to `print.data.frame`; `row.names` and `right` default to `FALSE` for the tidy look. For `format`, ignored. For `[`, the row and column indices and `drop`, as for a data frame. For `rbind`, the tables to combine and any options of `rbind.data.frame`, such as `make.row.names`.
`deparse.level`	Passed to `rbind()`.

Details

The stored numeric values are never rounded. Only the display changes, so any arithmetic you do on the returned object (a difference of two statistics, a further calculation) uses full precision. To see more digits, raise `digits` or read the column directly with `x$value`.

This mirrors the display convention of the **DMAR** package, whose house style **POEMED** follows.

Value

`print.poemed_tbl` returns `x` invisibly. `format.poemed_tbl` returns a `data.frame` whose numeric columns have been formatted to character for display. The `[` method returns a `poemed_tbl` with the attributes of `x` (or a vector, when a single column is extracted), and `rbind.poemed_tbl` returns a `poemed_tbl` whose attributes follow the rule under Subsetting and Combining.

Subsetting and Combining

A `poemed_tbl` carries its display information as attributes: which rows of the `value` column hold p-values, the outcome model, the active mediators identified, and the tuning record that the print

footer reports. Subsetting with `[` or `subset()` keeps all of them, whether you select rows, columns, or both, so a subset still prints its p-values with the floor and names the fit it came from. (Base R would drop them whenever columns are selected.)

Combining tables with `rbind()` keeps these attributes only when every argument is a `poemed_tbl` and all of them carry the same ones, as when you split one table and put it back together. Tables from different fits describe different models, so their combination keeps only the list of p-value rows and prints no single-fit footer. A row whose `term` begins with `pval_` prints as a p-value even when that list is missing.

Author(s)

Xiufan Yu and Ken Kelley

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 200, p = 60, pattern = "contrasting",
                             outcome = "continuous")
fit <- pe_mediation(d$X, d$Y, d$M, outcome = "continuous")
# HBIC chose the smallest value of the default grid here, so the fit
# warned; see the lambda_grid argument of ?pe_mediation.
fit                       # rounded for reading, with the p-value floor
fit$value[fit$term == "pval_pe"]   # full precision underneath

# A subset keeps the p-value format and the footer of its fit.
subset(fit, term %in% c("stat_pe", "pval_pe"))
```

`simulate_guo_mediation`

Simulate the Real-Data-Motivated Heterogeneous Mediation Setting

---

Description

Generates a linear mediation data set from the article's *real-data-motivated* heterogeneous setting, whose coefficients are calibrated to the DNA-methylation case study of Guo et al. (2022). Unlike the homogeneous and contrasting presets of `simulate_mediation_data()`, this setting has eleven active mediators with a genuine **mix of positive and negative** effects that neither all agree in sign (homogeneous) nor exactly cancel (contrasting): the messy middle ground that real data tends to present. The calibration constants are the guo_calibration object; the total indirect effect is `guo_calibration$beta_per_c1` times c_1, about $-1.597c_1$. (The article reports $-1.5977c_1$, from the full-precision Guo coefficients; the values shipped here are those coefficients rounded to three decimals.)

Usage

```
simulate_guo_mediation(
  c1,
  c2 = 0.5,
  confounders = FALSE,
  alpha_m = "setting0",
  n = NULL,
  seed = NULL
)
```

Arguments

c1	Signal-strength scale for the exposure-mediator coefficients (Gamma_x = c1 * Gamma_x_G), a single finite number. At c1 = 0 there is no mediation (the Type I error setting); the article sweeps c1 over c(0, +/- 0.1, ..., +/- 1).
c2	Direct effect (exposure-outcome coefficient), a single finite number. Default 0.5.
confounders	TRUE or FALSE; if TRUE, include the calibrated confounders in the data-generating process (the article's "with confounders" scenario). Default FALSE.
alpha_m	Outcome-mediator coefficients at the eleven active loci: either a numeric vector of 11 finite values, or one of the strings "setting0" (the default, the values 1, 0.9, 0.8, -0.9, -0.8, -0.7, 0.6, 0.5, 0.4, 0.3, and 0.2), "homogeneous_like", or "contrasting_like" (the two alternative sets the article also studies, which remain heterogeneous because they are paired with the scattered Gamma_x).
n	Number of observations, a single whole number of at least 2. Default NULL, the case-study size, 85.
seed	Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. NULL (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

Details

The mediator dimension ($p = 1008$), the eleven active loci, and the coefficient vectors come from Guo et al. (2022) and are fixed. The exposure and confounders, which the article took from the actual case-study data, are simulated here (standard normal) so the function is self-contained. The mediator and outcome coefficients, the part of the calibration that matters scientifically, are the values printed in the supplement of Yu and Kelley (in press), where the Guo estimates appear to three decimals.

Value

A list laid out like the value of simulate_mediation_data(), without its alpha_x, alpha_z, and Gamma_z components (the direct effect is c2, and the calibrated confounder coefficients are in guo_calibration): the exposure X (n by 1), mediators M (n by 1008), outcome Y, confounders Z (n by 8, or NULL), the full coefficient vectors alpha_m and Gamma_x, the total indirect effect beta, the active_mediators (the loci nonzero in both paths: all eleven under the default alpha_m, the first five under "homogeneous_like", the first four under "contrasting_like", and none when c1 = 0), and the dimensions.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

Guo, X., Li, R., Liu, J., & Zeng, M. (2022). High-dimensional mediation analysis for selecting DNA methylation loci mediating childhood trauma and cortisol stress reactivity. *Journal of the American Statistical Association, 117*(539), 1110–1121. doi:10.1080/01621459.2022.2053136

See Also

`simulate_mediation_data()`, guo_calibration, `pe_power_curve()`.

Other mediation simulation: `guo_calibration`, `mediation_ar1_cov()`, `pe_identification_study()`, `pe_power_curve()`, `pe_simulation_study()`, `plot.poemed_tbl()`, `simulate_mediation_data()`, `ss_power_pe_mediation()`

Examples

```
set.seed(113)
d <- simulate_guo_mediation(c1 = 0.5)
dim(d$M)               # 85 x 1008
d$active_mediators     # the 11 active loci
round(d$beta, 4)       # beta_per_c1 * 0.5, i.e. -0.7985
# On the draw above (c1 = 0.5 at the case study's n = 85) neither test
# rejects. With c1 = 1 and n = 150 both tests reject the global null, and
# the screen identifies seven of the eleven truly active loci. The fit
# searches a 20-value grid to keep the example fast; on the default grid
# the screen identifies the same seven loci and the fit takes about five
# times as long.
d2 <- simulate_guo_mediation(c1 = 1, n = 150, seed = 113)
fit <- pe_mediation(d2$X, d2$Y, d2$M, outcome = "continuous",
                    lambda_grid = seq(0.05, 2, length.out = 20))
fit[fit$term %in% c("pval_hdmm", "pval_pe", "n_active_mediators"), ]
```

---

simulate_mediation_data

Simulate High-Dimensional Mediation Data

Description

Generates a data set from the mediation models studied in the article, for a continuous, binary, or count outcome, under the homogeneous or contrasting (heterogeneous) mediation patterns. The mediators carry an AR(1) correlation structure, a handful are truly active, and the rest are null, so the data exercise exactly the situation the power-enhanced tests are designed for. This is the generator behind the package's example data sets and behind `pe_power_curve()`.

Usage

```
simulate_mediation_data(
  n,
  p,
  outcome = c("continuous", "binary", "count"),
  pattern = c("homogeneous", "contrasting", "heterogeneous"),
  c1 = 1,
  c2 = 0.5,
  q = 1L,
  d = 0L,
  alpha_m = NULL,
  tau = NULL,
  rho = 0.5,
  sigma_y = 0.5,
  alpha_z = NULL,
  Gamma_z = NULL,
  seed = NULL
)
```

Arguments

`n`	Number of observations.
`p`	Number of candidate mediators.
`outcome`	Outcome type: `"continuous"`, `"binary"`, or `"count"`.
`pattern`	Mediation pattern: `"homogeneous"` (active effects share a sign) or `"contrasting"` (active effects of opposite sign). `"heterogeneous"` is accepted as a synonym for `"contrasting"`. Ignored when `alpha_m` is supplied directly. The presets are the article's. With the default `tau`, the contrasting presets for the continuous and binary outcomes cancel to a total indirect effect of zero, while the count preset, (0, 0, 0, 0.8, -0.7), gives a total indirect effect of -0.03 times `c1`.
`c1`	Scale on the exposure-on-mediator coefficients $\Gamma_x = c_1\tau$. `c1 = 0` makes every mediator inactive (the global null). Default 1.
`c2`	The direct effect α_x of each exposure on the outcome. Default 0.5.
`q`	Number of exposures. Default 1. For `q > 1` you must supply `tau` as a `q` by `p` matrix.
`d`	Number of confounders. Default 0 (none). When positive, confounders are drawn standard normal and enter both the mediator and the outcome model through `Gamma_z` and `alpha_z`, whose coefficients are themselves drawn from U(-0.5, 0.5) unless supplied. The article's "with confounders" scenario is `d = 5`.
`alpha_m`	Optional numeric vector of `p` finite mediator-on-outcome coefficients, overriding the `pattern` preset.
`tau`	Optional base pattern for Γ_x, finite numbers. By default it is the article's (0.1, 0.2, 0.3, 0.4, 0.5, noise), for `q = 1`, with the noise loadings drawn from N(0, 0.1^2) for a continuous outcome and N(0, 0.01^2) for a binary or count one, the scales of the article's respective Monte Carlo designs. Required when `q > 1`, as

a q by p matrix (one row per exposure) or a vector of its values in column order; when q = 1 a vector of length p serves.

rho — AR(1) correlation parameter for the mediator noise. Default 0.5.

sigma_y — Outcome noise standard deviation for a continuous outcome. Default 0.5.

alpha_z, Gamma_z — Optional confounder coefficients: a vector of d finite numbers and a d by p matrix (when d = 1, a vector of length p serves for Gamma_z). When d > 0 and either is left NULL, it is drawn from U(-0.5, 0.5), the article's Monte Carlo design, afresh on every call. Gamma_z is then drawn as d values recycled across the p mediator columns, reproducing the article's code: each confounder loads equally on every mediator, so the confounders enter the mediators as a single common factor. Supply Gamma_z yourself for d by p independent loadings. Both are used only when d > 0; supplying either with d = 0, where it would be ignored, is an error.

seed — Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. NULL (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

Details

One exposure column (or q of them) is drawn standard normal. The mediators follow $M = X\Gamma_x + Z\Gamma_z + \varepsilon_m$ with $\varepsilon_m \sim N(0, \Sigma)$, Σ the AR(1) covariance mediation_ar1_cov() with parameter rho. The exposure-on-mediator coefficient is $\Gamma_x = c_1\tau$, where the base pattern τ has small increasing loadings on the first five mediators and random noise loadings on the rest, so c1 scales the overall exposure-to-mediator signal (and c1 = 0 gives the global null of no mediation). The outcome is generated from its model with mediator coefficients α_m (the preset for the chosen pattern, or a user-supplied vector) and direct effect $\alpha_x = c_2$:

- continuous: $Y = M\alpha_m + X\alpha_x + Z\alpha_z + \varepsilon_y$, $\varepsilon_y \sim N(0, \sigma_y^2)$;
- binary: $Y \sim \text{Bernoulli}(\text{logit}^{-1}(M\alpha_m + X\alpha_x + Z\alpha_z))$;
- count: $Y \sim \text{Poisson}(\exp(\eta))$, where the log-mean η passes through the article's overflow guard: if its largest value exceeds 5, the whole vector is mapped affinely onto the interval from -5 to 5 before exponentiating, and otherwise it is left alone. The guard tests only the maximum, so a run of very negative values passes through untouched.

With d > 0 the confounders are standard normal and enter the mediators as $Z\Gamma_z$ and the outcome as $Z\alpha_z$, with both coefficient sets drawn from U(-0.5, 0.5) unless given. They leave the total indirect effect alone, since that runs through Γ_x and α_m, but they correlate the mediators with each other and with the outcome, which is what makes adjusting for them necessary.

Every design argument is checked before any random number is drawn: a coefficient vector or matrix of the wrong length or shape, or one holding a missing or infinite value, stops with a message naming it.

Value

A list with the simulated data and the truth used to generate it: X (n by q), M (n by p), Y (length n), Z (n by d, or NULL), the coefficient values alpha_m, Gamma_x, alpha_x, the confounder coefficients

alpha_z and Gamma_z that were used (NULL when d = 0, so a drawn pair is recoverable), the total indirect effect beta (Gamma_x times alpha_m), and active_mediators (the indices of the truly active mediators, those nonzero in both paths), together with n, p, q, outcome, and pattern.

For a count outcome beta is the value before the overflow guard described under Details. When that guard fires, it rescales the log-mean by a factor the sample itself determines, so the coefficients realized in that replication, and with them the total indirect effect, are beta times that factor rather than beta.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

pe_mediation() to analyze the data, pe_power_curve() to sweep c1, mediation_ar1_cov() for the mediator covariance.

Other mediation simulation: guo_calibration, mediation_ar1_cov(), pe_identification_study(), pe_power_curve(), pe_simulation_study(), plot.poemed_tbl(), simulate_guo_mediation(), ss_power_pe_mediation()

Examples

```
set.seed(113)
d <- simulate_mediation_data(n = 100, p = 50, outcome = "continuous",
                             pattern = "contrasting")
dim(d$M)
d$active_mediators       # truly active mediators
round(d$beta, 10)        # total indirect effect, zero up to rounding error
```

---

ss_power_pe_mediation *Simulation-Based Sample Size Planning for the Power-Enhanced Mediation Test*

Description

Estimates, by Monte Carlo simulation, the sample size needed for the power-enhanced test to reach a target power under a given mediation pattern and signal strength. It evaluates the empirical power of the PE test (and, for comparison, the benchmark Wald test) at each candidate sample size on a grid, and reports the smallest grid value that reaches the target. This is the design counterpart of the analysis function pe_mediation(): use it to plan a study, then pe_mediation() to analyze it.

Usage

```
ss_power_pe_mediation(
  outcome = c("continuous", "binary", "count"),
  pattern = c("homogeneous", "contrasting", "heterogeneous"),
  p,
  n_grid,
  c1 = 1,
  c2 = 0.5,
  target_power = 0.8,
  n_rep = 200,
  alpha_level = 0.05,
  method = c("Bonferroni", "BH", "BY"),
  error_level = 0.05,
  lambda_grid = seq(0.05, 10, length.out = 100),
  cores = 1L,
  seed = NULL
)
```

Arguments

`outcome` Outcome type: `"continuous"`, `"binary"`, or `"count"`.

`pattern` Mediation pattern: `"homogeneous"` or `"contrasting"` (`"heterogeneous"` is a synonym for `"contrasting"`).

`p` Number of candidate mediators.

`n_grid` Vector of candidate sample sizes to evaluate: whole numbers of at least 10 (a value listed twice is evaluated once). They are evaluated, and reported, in increasing order.

`c1` Signal strength (the exposure-on-mediator scale), a single finite nonzero number; at `c1 = 0` the rejection rate is the test's size, which no sample size raises to a power target. Default 1.

`c2` Direct effect. Default 0.5, the value of the article's linear simulations; its logistic figures use 1 and its Poisson figures 0.4.

`target_power` Desired power for the PE test, a single number between 0 and 1 (see Details for the replications it needs). Default 0.8.

`n_rep` Monte Carlo replications per candidate `n`. Default 200.

`alpha_level` Significance level. Default 0.05.

`method, error_level` Passed to `pe_mediation()`.

`lambda_grid` Passed to `pe_mediation()`: the tuning grid searched at each candidate `n`, by default `seq(0.05, 10, length.out = 100)`. A plan runs `n_rep` fits at every candidate `n`, so a shorter or narrower grid speeds it up.

`cores` Number of CPU cores (Unix forking). A seeded parallel run is reproducible across runs at the same `cores` but need not match a serial run. Default 1.

seed — Optional integer seed. When supplied it is set for the duration of the call and the caller's random number generator state is restored on exit. NULL (the default) sets no seed: the draws come from the session's random number stream, which the call advances as any random function does.

Details

This is a simulation-based procedure, not a closed-form power formula: no analytic power expression exists for the PE test, so sample size is instead determined by direct Monte Carlo simulation. Power is estimated by simulating n_rep data sets at each candidate n with simulate_mediation_data() and recording how often the test rejects at level alpha_level. Because the estimate is a Monte Carlo proportion, it carries simulation error of roughly $\sqrt{power(1-power)/n_rep}$; raise n_rep for a smoother curve and a more stable recommendation. The grid approach (rather than a root search) is deliberate: the power curve is monotone but noisy, so a search can stop early on a lucky draw, whereas the grid shows the whole trajectory.

An estimated power from n_rep replications moves in steps of 1 / n_rep, so a target above 1 - 1 / n_rep can be reached only when every replication rejects, which is weak evidence that the true power reaches it. Such a target draws a warning of class poemed_target_unresolvable naming the smallest n_rep that resolves it. When no candidate sample size reaches the target, the recommendation is NA and a warning of class poemed_target_not_reached gives the largest estimated power and where it occurred, so the grid can be extended. The printed table ends with the target and the recommendation (or the statement that the grid did not reach the target).

Value

A tidy data.frame (class poemed_ss_plan, a poemed_tbl) with one row per candidate sample size and columns n, power_pe, power_hdmm, n_valid, and n_empty (see pe_power_curve()). The smallest n reaching target_power for the PE test is stored in the "recommended_n" attribute (NA if no grid value reaches it), with "target_power", "outcome", "pattern", "c1", "c2", "n_rep", and "alpha_level". Printing the table adds a line with the target and the recommendation.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. citation("POEMED") gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

See Also

pe_power_curve(), pe_mediation(), simulate_mediation_data().

Other mediation simulation: guo_calibration, mediation_ar1_cov(), pe_identification_study(), pe_power_curve(), pe_simulation_study(), plot.poemed_tbl(), simulate_guo_mediation(), simulate_mediation_data()

Examples

```
# n_rep = 5 keeps this example fast; a power estimate from five
# replications has a Monte Carlo standard error of up to 0.22, so the
# recommended n here is a demonstration, not a plan. Planning a study
# deserves n_rep of several hundred. A plan is also where a shorter
# tuning grid pays: every replication searches the whole grid, so this
# one passes 20 values from 0.05 to 2 instead of the default 100 values
# from 0.05 to 10 (see the lambda_grid argument of ?pe_mediation).
set.seed(113)
plan <- ss_power_pe_mediation(outcome = "continuous", pattern = "contrasting",
                              p = 50, n_grid = c(150, 250), n_rep = 5,
                              lambda_grid = seq(0.05, 2, length.out = 20))
plan                        # the last line gives the recommendation
attr(plan, "recommended_n") # 150, the smallest grid n reaching 0.8
```

---

summary.poemed_tbl *Summarize a POEMED Table*

Description

For a grouped table that compares the benchmark and power-enhanced tests (any poemed_tbl with a group column, a pval_hdmm column, and a power-enhanced p-value column (pval_pe or pval_pe_bonferroni), such as the output of WHO_mediation_analysis()), summary() digests it across groups: how many groups each test detects mediation in, and in particular the groups where the power-enhanced test detects mediation that the benchmark misses, together with the most frequently flagged mediators. For any other poemed_tbl it falls back to the ordinary data-frame summary.

Usage

```
## S3 method for class 'poemed_tbl'
summary(object, alpha_level = 0.05, ...)

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

Arguments

object	A poemed_tbl.
alpha_level	Significance level for counting a group as a detection. Default 0.05.
...	For a grouped comparison table, ignored. For any other poemed_tbl, passed to summary.data.frame() (for example digits or quantile.type).
x	A summary.poemed_tbl object, for the print method.

Details

The printed digest follows the display convention of `print.poemed_tbl()`: p-values appear to four decimal places, with a `< 0.0001` floor and never in scientific notation, and the significance level appears as supplied. The returned object keeps the p-values at full precision.

Value

For a grouped comparison table, an object of class `summary.poemed_tbl` (a list, printed by its own method) with elements `outcome` (the outcome label, or `NULL` when the table has none), `full_table` (logical; `TRUE` for the extended layout of `WHO_mediation_analysis()` with `full_table = TRUE`), `alpha_level`, `n_groups` (rows in the table), `n_with_data` (groups with a benchmark p-value), `n_hdmm` and `n_pe` (groups detected by each test), `pe_only` (a `data.frame` of the groups the PE test detects but the benchmark does not, with their `n_countries`, both p-values, and the `active_mediators` codes), and `top_mediators` (a frequency table of the mediators identified as active in the groups the PE test detects). For any other `poemed_tbl`, the value of `summary.data.frame()`. The print method returns its argument invisibly.

Author(s)

Xiufan Yu and Ken Kelley

See Also

`WHO_mediation_analysis()`, `pe_mediation()`.

Other mediation tests: `pe_mediate()`, `pe_mediation()`, `pe_mediation_linear()`, `pe_mediation_logistic()`, `pe_mediation_poisson()`, `pe_mediators()`, `pe_selection()`

Examples

```
# The global and income-group infant-mortality models, on a 25-point
# version of the default 100-point grid to keep the example fast (the
# global model's grid defaults to the same grid). On the coarser grid
# the screen identifies the same active mediators as on the default grid,
# those of the article's Table 1, and some benchmark p-values move
# slightly.
res <- WHO_mediation_analysis("imr", groupings = c("global", "income"),
                              lambda_grid = seq(0.1, 2, length.out = 25))
summary(res)
```

`WHO_health_mediation` *WHO / World Bank Health-Expenditure Mediation Data (Benchmark Data Set)*

Description

A benchmark data set for high-dimensional mediation methods: the World Health Organization (WHO) and World Bank panel analyzed in Yu and Kelley (in press). It is a country-by-year panel used to ask how health-care expenditure mediates the relationship between economic growth and population health outcomes. The exposure is economic growth, the candidate mediators are 57 health-expenditure indicators, and five health outcomes are available. With 57 intercorrelated mediators this is a genuinely high-dimensional mediation problem of the kind the power-enhanced tests are built for, and the mediation effects are heterogeneous (different kinds of spending push different outcomes in different directions), which is exactly where the benchmark test loses power and the PE test does not.

Usage

```
WHO_health_mediation
```

Format

`WHO_health_mediation` is a `data.frame` with 2002 rows (91 WHO member states times the years 2000–2021) and 68 columns:

code ISO-3166 alpha-3 country code.

country Country name.

region WHO region: `AFR` (Africa, 32 members), `AMR` (Americas, 28), `EMR` (Eastern Mediterranean, 8), `EUR` (Europe, 9), `SEAR` (South-East Asia, 5), `WPR` (Western Pacific, 9).

income World Bank income group, a factor (not an ordered one) whose levels run from low to high: `Low` (16 members), `Lower-middle` (27), `Upper-middle` (26), `High` (22). The article abbreviates these Low, LM, UM, High.

year Calendar year, 2000–2021.

gdp_growth The exposure X: annual percentage growth of GDP per capita (World Bank Open Data), a common proxy for economic growth.

imr Infant mortality rate (deaths per 1000 live births).

u5mr Under-five mortality rate (deaths per 1000 live births).

leb Life expectancy at birth (years).

lbw Prevalence of low birthweight (percent of live births; covers 73 members, 2000–2020, so other rows are `NA`).

pou Prevalence of undernourishment (percent of population; covers 79 members, 2001–2021, so other rows are `NA`).

che_gdp, ..., ext_usd2021_pc The 57 candidate mediators M: health-expenditure indicators from the WHO Global Health Expenditure Database. Their codes and full names are in WHO_indicator_codebook. They are complete (no missing values).

Details

It is shipped as a benchmark so that a method, old or new, can be evaluated on a real, published high-dimensional mediation problem (see `WHO_mediation_analysis()`, whose output is the package's

benchmark fit). The World Health Organization (WHO) is the source of the health-expenditure mediators and the region and income labels; the World Bank is the source of the economic-growth exposure and the health outcomes.

Three of the five health outcomes (`imr`, `u5mr`, `leb`) cover all 91 members for 2000–2021; `lbw` and `pou` have narrower coverage, so they carry `NA` where unavailable. The mediators are intercorrelated by construction, since many are different normalizations of the same underlying spending (per capita, percent of GDP, percent of current health expenditure, constant versus current currency, and so on), which is precisely the high-dimensional, correlated-mediator regime the methods target.

Use `WHO_mediation_design()` to assemble the exposure, outcome, mediator, and confounder matrices for a given outcome (and optionally a region or income subset) ready for `pe_mediation()`. In the article's global model the confounders are the region, income group, and year.

Source

Exposure and health outcomes: World Bank Open Data, World Development Indicators (https://data.worldbank.org/), licensed by the World Bank under the Creative Commons Attribution 4.0 International license (CC BY 4.0, https://datacatalog.worldbank.org/public-licenses), which permits redistribution and adaptation with attribution and a note of the changes made. Health-expenditure mediators and the region and income labels: World Health Organization, Global Health Expenditure Database (https://apps.who.int/nha/database), made available under WHO's terms and conditions for the use of its data (https://www.who.int/about/policies/publishing/data-policy/terms-and-conditions): non-commercial use for public health purposes, with attribution to WHO and to the countries that provided the underlying data, and no suggestion that WHO endorses any product or service. Attribution: World Health Organization, Global Health Expenditure Database, accessed November 2024; World Bank, World Development Indicators, accessed November 2024. Changes made: the two sources were merged by country and year, restricted to the 91 WHO member states with complete health-expenditure data for 2000 to 2021, and the outcome series were attached; no value was altered. The script that builds the data set, `data-raw/WHO_health_mediation.R`, is in the package's GitHub repository (https://github.com/yelleKneK/POEMED), not in the package itself. The package's `LICENSE.note` file restates these terms.

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

`WHO_mediation_design()` to build an analysis design, WHO_indicator_codebook for the mediator definitions, `pe_mediation()` to run the test.

Other WHO mediation data: `WHO_indicator_codebook`, `WHO_mediation_analysis()`, `WHO_mediation_design()`

Examples

```
data(WHO_health_mediation)
table(WHO_health_mediation$region[WHO_health_mediation$year == 2010])

# A first test on the global infant-mortality design with the package
```

```
# defaults, which standardize X, M, and Z and center Y.
# WHO_mediation_analysis() instead applies the preprocessing and the
# tuning grid behind the article's tables, which leaves the
# confounders unscaled (see ?WHO_mediation_design for the difference).
# HBIC chooses 0.05, the smallest value of the default grid, so this fit
# warns; WHO_mediation_analysis() searches the article's grid instead.
des <- WHO_mediation_design("imr")
pe_mediation(des$X, des$Y, des$M, Z = des$Z, outcome = "continuous")
```

`WHO_indicator_codebook`

Codebook for the WHO Health-Expenditure Indicators

Description

The 57 health-expenditure indicators that serve as candidate mediators in WHO_health_mediation, with their short codes and descriptions. The codes and their order are those of Table S.7 in the supplement to Yu and Kelley (in press); the descriptions are adapted from that table rather than copied, abbreviating some spelled-out denominators (for example "as % of GDP" for "as % of Gross Domestic Product (GDP)" and "as % of CHE" for "as % of Current Health Expenditure (CHE)"). Many indicators are alternative normalizations of the same underlying spending (per capita, percent of GDP, percent of current health expenditure, current versus constant 2021 currency, national currency versus US dollars versus purchasing power parity), which is why they are strongly intercorrelated.

Usage

```
WHO_indicator_codebook
```

Format

A `data.frame` with 57 rows and 2 columns:

indicator The short indicator code, matching a column name in WHO_health_mediation (for example `gge_gdp`, `shi_che`).

description The full indicator name. Common abbreviations: CHE (current health expenditure), GGHE-D (domestic general government health expenditure), PVT-D (domestic private health expenditure), EXT (external health expenditure), OOPS (out-of-pocket spending), SHI (social health insurance), GGE (general government expenditure), GDP (gross domestic product), NCU (national currency unit), PPP (purchasing power parity).

Source

Adapted from Table S.7 ("A List of Health Spending Indicators and the Associated Indicator Code used in Real Data Analysis") of the supplement to Yu and Kelley (in press); underlying definitions from the WHO Global Health Expenditure Database.

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

WHO_health_mediation, `WHO_mediation_design()`.

Other WHO mediation data: `WHO_health_mediation`, `WHO_mediation_analysis()`, `WHO_mediation_design()`

Examples

```
data(WHO_indicator_codebook)
head(WHO_indicator_codebook)
# Look up a mediator flagged in the article's analysis.
WHO_indicator_codebook[WHO_indicator_codebook$indicator == "gge_gdp", ]
```

`WHO_mediation_analysis`

Run the Article's Empirical Mediation Analysis

Description

Runs the article's real-data analysis for one health outcome across the groupings it reports: a global model using all WHO members, then separate models within each WHO region, within each World Bank income group, and (optionally) within each region-by-income cell. For every grouping it reports the benchmark Wald test and the power-enhanced test on the total indirect effect of health expenditure, together with the individual mediators the PE screen identifies as active. The result has the layout of the article's data-analysis tables (Table 1 for infant mortality and Table 2 for undernourishment; Tables S.8 to S.12 of its supplement give the extended layout and the other three outcomes), computed from the shipped WHO_health_mediation data.

Usage

```
WHO_mediation_analysis(
  outcome = c("imr", "u5mr", "leb", "lbw", "pou"),
  groupings = c("global", "region", "income"),
  error_level = 0.05,
  lambda_grid = seq(0.1, 2, length.out = 100),
  lambda_grid_global = lambda_grid,
  data = WHO_health_mediation,
  cores = 1L,
  verbose = FALSE,
  full_table = FALSE
)
```

Arguments

- `outcome` — Which health outcome to analyze: `"imr"`, `"u5mr"`, `"leb"`, `"lbw"`, or `"pou"`. See WHO_health_mediation.
- `groupings` — Character vector of groupings to run, any of `"global"`, `"region"`, `"income"`, and `"region_income"`. Default `c("global", "region", "income")` (the region-by-income cells are the slowest and are opt-in).
- `error_level` — Target familywise error rate for the mediator-screening step. Default 0.05, as in Yu and Kelley (in press).
- `lambda_grid` — Tuning-parameter grid for the subgroup penalized fits (regional, income, and region-by-income models). Default `seq(0.1, 2, length.out = 100)`, the grid behind the article's real-data tables (Tables 1, 2, and S.8 to S.12 of Yu and Kelley, in press); it differs from the default of `pe_mediation()` because the function reproduces the article's analysis as run. Each fit searches it for the full model and for the reduced model alike; the function takes no separate `lambda_grid_reduced`.
- `lambda_grid_global` — Tuning-parameter grid for the global (all-country) model only. Defaults to `lambda_grid`: the article's tables use the same grid for the global model and the subgroup models.
- `data` — The source data, in the shape of WHO_health_mediation; see `WHO_mediation_design()` for the columns it must carry. Defaults to WHO_health_mediation.
- `cores` — Number of CPU cores; values above 1 fit the subgroups in parallel via the base **parallel** package (Unix only). Default 1.
- `verbose` — Logical; if `TRUE`, print each group as it is fit. Default `FALSE`.
- `full_table` — Logical; if `TRUE`, return the article's *extended* table layout, with the power-enhanced p-value and the codes of the active mediators identified reported separately under each multiplicity method (Bonferroni for FWER, BH and BY for FDR) instead of a single primary method. Default `FALSE`.

Details

The output is the package's **benchmark fit**: the test suite pins the identified active sets and the benchmark p-values of the article's tables under the function's default grid, so a change to the estimation code that moved them would be caught, and the fit serves as a fixed reference point for the method.

Each subgroup is fit with the preprocessing of the analysis behind the article's tables: the exposure and the 57 mediators are standardized, the outcome is centered, and the confounders (the region and income indicators and the year) are left unscaled. The article's text says that covariates are standardized; the unscaled confounders are a choice of that analysis. Within a group, an indicator that is constant there (common in the small region-by-income cells) is dropped before fitting, since it carries no information. An indicator with a missing or non-finite value among the group's rows is dropped as well, because the method has no missing-data mechanism; those are named in a message, since they change the set of candidate mediators.

Four kinds of event are reported once, after all groups are fit, and recorded in the `"notes"` attribute (a `data.frame` with `group`, `stage`, and `message`, or `NULL` when there are none). The `stage` column says which:

- "design": a group with too few complete observations (some region-by-income cells are empty or nearly so), or whose rows lack a country code, a year, or a region or income group that varies within the group, is skipped and reported as an NA row, named in a message.
- "missing": indicators with missing or non-finite values were dropped in the group before fitting, named in a message.
- "empty": the penalized fit that HBIC selected contains no mediator. The row keeps the values of POEMED's reporting convention for an empty selection (p-values of 1, no active mediator identified), which are not evidence for the null, and the group is named, with the lambda HBIC selected, in a warning of class poemed_empty_fit at any value of cores.
- "fit": the fit failed; the group is reported as an NA row and named in a warning that quotes the error.

Because every group fits a penalized model over a grid of tuning parameters, a full run with all groupings fits dozens of models and takes about ten seconds on a current laptop; start with the default groupings, or a single outcome, before scaling up. In 67 of the 128 cells that the five outcomes fit, the HBIC choice for the full penalized model or for the reduced model of the benchmark Wald test (54 cells each) is 0.1, the grid's smallest value; that is a property of the article's analysis on its fixed grid, so the function does not raise the single-fit poemed_grid_boundary warning for it. To see a cell's tuning record, fit the cell by hand with pe_mediation_linear(), as the reproduction vignette shows.

The tuning-parameter grid is seq(0.1, 2, length.out = 100) for the global model and for every subgroup model, the grid behind the article's Tables 1, 2, and S.8 to S.12 (Section 5 of Yu and Kelley, in press); the article's text itself states only that the tuning parameter is chosen by the HBIC criterion. The article's real-data analysis was run with this package (pe_mediation_linear() at that grid, with the preprocessing described above); WHO_mediation_analysis() automates those calls. Under this default the function reproduces those tables. The identified active set agrees with the printed one in every fitted cell under Bonferroni, BY, and BH, with one exception (the BH set for under-five mortality in the High income group, where the article's set also holds gge_gdp). The benchmark p-value agrees to four decimals in all but a handful of cells: infant mortality in SEAR (0.6483 here, 0.6482 printed) and in AMR-High (0.0726 here, 0.0725 printed) differ by rounding in the fourth decimal, and a few p-values at or below 0.0001, which the article prints to one significant figure, come out somewhat smaller here. Exact p-values depend on the grid and on the unscaled confounders. Six of the region-by-income cells that the article's tables fill are not fit at all and are reported as NA (the cells the tables list with 0 countries are NA too, skipped before fitting):

- four cells in which two indicators are exact linear partners and the fit keeps both, so a matrix it inverts is singular: infant mortality and under-five mortality in EMR-Lower-middle (dom_che and ext_che sum to 100 there), and life expectancy and under-five mortality in AFR-Upper-middle (ext_che with dom_che, and shi_che with chi_che). The article's values are reproduced by dropping one partner; see the recipe below.
- two cells of the undernourishment outcome, EUR-High and WPR-High, in which the outcome is 2.5 in every country-year (the lower reporting bound of the source), so there is no variation to mediate. The function reports the fit as failed with the message that Y is constant; the article prints a p-value of 1 there.

Many of the 57 indicators are exact or near-exact linear combinations of one another, and the penalized selection chooses among such partners. As a result the benchmark p-values, and occasionally the active mediators identified, can change when the indicator columns of data are reordered, even

between two indicators that are exactly collinear within a group. The results reported are those for the column order of the shipped data.

A group whose fit fails with a singular-matrix error usually holds two indicators that are exact linear partners within that group (for example, `dom_che` and `ext_che` sum to 100 in the Eastern Mediterranean lower-middle-income group; `stats::cor()` of 1 or -1 among the columns of the group's `M` finds such pairs). Such a group can be fit by hand after dropping one member of the pair. Which member is dropped matters, since the two fits select among different columns (dropping `ext_che` instead in the infant-mortality cell below gives 0.0786). The article's analysis removed `dom_che` and `vfa_che` in the two EMR-Lower-middle cells (`cfa_che` and `vfa_che` are near-exact partners there as well), giving benchmark p-values of 0.0837 for infant mortality and 0.0914 for under-five mortality with no active mediator identified, and `shi_che` and `ext_che` in the two AFR-Upper-middle cells, giving 0.1311 for life expectancy with no active mediator identified and 0.0389 for under-five mortality with `pvtd_ncu2021_pc` identified as active. With the default grid, the infant-mortality cell is:

```
cell <- WHO_mediation_design("imr", region = "EMR",
                             income = "Lower-middle")
keep <- apply(cell$M, 2, sd) > 0 &
  !(cell$indicators
grid <- seq(0.1, 2, length.out = 100)
pe_mediation(scale(cell$X), cell$Y - mean(cell$Y),
             scale(cell$M[, keep]), Z = cell$Z,
             outcome = "continuous", scale = FALSE,
             lambda_grid = grid)
```

which prints the benchmark p-value 0.0837 of the article's Table 1.

Value

A `data.frame` with one row per fitted subgroup. With `full_table = FALSE` (the default) the columns are `grouping` (global / region / income / region_income), `group` (the specific group, for example `AFR` or `Low`), `n_countries` (number of countries; `NA` for a group skipped at the design stage: one with fewer than 10 complete observations, which includes every group the article's tables list with 0 countries, or one whose rows lack a key the group needs), `pval_hdmm` and `pval_pe` (the benchmark and power-enhanced p-values for the total indirect effect), `n_active` (number of active mediators the screen identified), and `active_mediators` (their codes, or `"none"`). With `full_table = TRUE` the single `pval_pe`/`n_active`/`active_mediators` columns are replaced by `pval_pe_bonferroni`, `pval_pe_bh`, `pval_pe_by` and the matching `active_bonferroni`, `active_bh`, `active_by` code columns. The result is a `poemed_tbl` with the attributes `"outcome"`, `"full_table"`, and `"notes"` (see Details); call summary() on it for a cross-group digest of where the power-enhanced test detects mediation the benchmark misses.

How to Cite

If you use POEMED in published work, please cite Yu and Kelley (in press), the article that introduces its methods and the package. `citation("POEMED")` gives the full reference and a BibTeX entry.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

`WHO_mediation_design()` for a single design, WHO_health_mediation for the data, `pe_mediation()` for the test.

Other WHO mediation data: `WHO_health_mediation`, `WHO_indicator_codebook`, `WHO_mediation_design()`

Examples

```
# The global infant-mortality model, the first row of the article's
# Table 1 (under a second): benchmark p-value 0.0644, power-enhanced
# p-value below 0.0001, and gge_gdp identified as active. Add "region"
# and "income" to groupings for the rest of the table.
WHO_mediation_analysis("imr", groupings = "global")
```

`WHO_mediation_design` *Build a Mediation Design From the WHO Health-Expenditure Data*

Description

Assembles the exposure, outcome, mediator, and confounder matrices for one of the analyses in Yu and Kelley (in press) from WHO_health_mediation, ready to pass to `pe_mediation()`. The exposure is GDP per capita growth, the mediators are the 57 health-expenditure indicators, and the outcome is the chosen health indicator. Optionally restrict to a WHO region and/or a World Bank income group, as in the article's region-specific, income-specific, and region-by-income models.

Usage

```
WHO_mediation_design(
  outcome = c("imr", "u5mr", "leb", "lbw", "pou"),
  region = NULL,
  income = NULL,
  data = WHO_health_mediation
)
```

Arguments

`outcome` Which health outcome to use as Y: `"imr"` (infant mortality rate), `"u5mr"` (under-five mortality rate), `"leb"` (life expectancy at birth), `"lbw"` (prevalence of low birthweight), or `"pou"` (prevalence of undernourishment).

`region` Optional character vector of WHO regions to keep (`"AFR"`, `"AMR"`, `"EMR"`, `"EUR"`, `"SEAR"`, `"WPR"`). Default `NULL` keeps all regions.

`income` Optional character vector of income groups to keep (`"Low"`, `"Lower-middle"`, `"Upper-middle"`, `"High"`). Default `NULL` keeps all groups.

`data` The source data frame, in the shape of WHO_health_mediation: the columns `code`, `region`, `income`, `year`, `gdp_growth`, and the chosen outcome, plus at least two indicator columns. Every column other than the key, exposure, and outcome columns of WHO_health_mediation is treated as a candidate mediator, so the year, the exposure, the outcome, and every indicator column must be numeric; `region` and `income` may be character or factor. Every row the design uses needs its country `code` and `year`, and its `region` and `income` wherever those vary among the rows kept (that is, wherever they enter the confounders); a row missing one of them stops the design with a message naming the country. Rows outside the requested region or income group, or dropped for a missing outcome or exposure, are not checked. A `data` that breaks these rules stops with a message naming the column. Defaults to WHO_health_mediation.

Details

Rows with a missing outcome or missing exposure are dropped (the article handles each outcome's coverage separately, which is why the low-birthweight and undernourishment outcomes have fewer observations). The confounder matrix Z contains the WHO region and income group (as indicator variables) and the year (numeric), the covariate set of the article. The region and income indicators enter only while they still vary after subsetting: a region-specific model drops the (now constant) region, and unused factor levels are dropped so no all-zero indicator columns remain. The year always enters, so a subset must span at least two years; a single-year subset leaves a constant year column, which `pe_mediation()` rejects.

The indicator columns are passed through as they are. A missing value in one of them makes `pe_mediation()` stop, since the method has no missing-data mechanism; `WHO_mediation_analysis()` instead drops such an indicator within the affected group and names it in a message.

To fit with the preprocessing of the analysis behind the article's tables, standardize the exposure and mediators and center the outcome while leaving the confounders (the region and income indicators and the year) unscaled. The article's text says that covariates are standardized; the unscaled confounders are a choice of that analysis. The grid below, `seq(0.1, 2, length.out = 100)`, is the grid behind the article's Tables 1, 2, and S.8 to S.12 and the default of `WHO_mediation_analysis()`; the call gives the first row of Table 1 (benchmark p-value 0.0644, gge_gdp identified as active):

```
des <- WHO_mediation_design("imr")
pe_mediation(scale(des$X), des$Y - mean(des$Y), scale(des$M),
             Z = des$Z, outcome = "continuous", scale = FALSE,
             lambda_grid = seq(0.1, 2, length.out = 100))
```

HBIC chooses 0.1, the smallest value of that grid, for this fit, as it does in about half of the article's cells, so the fit warns (class `poemed_grid_boundary`); `WHO_mediation_analysis()` keeps the article's grid and does not raise the warning (see its Details).

The simpler call `pe_mediation(des$X, des$Y, des$M, des$Z, outcome = "continuous")` standardizes X, M, and Z and centers Y, and so fits a different model: scaling the confounder columns changes both the benchmark Wald test and the penalized selection, so its p-values and its identified active set can differ from those of the call above. `WHO_mediation_analysis()` applies the preprocessing and the grid of the article's analysis for you.

Value

A list with `X` (exposure, $n \times 1$), `Y` (outcome vector), `M` (the $n \times 57$ mediator matrix), `Z` (the confounder matrix: the region and income indicators that still vary after subsetting, and the year), and the metadata `outcome`, `n` (country-year observations), `n_countries` (unique countries), `indicators`, `region`, and `income`.

Author(s)

Xiufan Yu and Ken Kelley

References

Yu, X., & Kelley, K. (in press). Power Enhancement in High-Dimensional Heterogeneous Mediation Analysis. *Journal of the American Statistical Association*.

See Also

`pe_mediation()`, WHO_health_mediation, WHO_indicator_codebook.

Other WHO mediation data: `WHO_health_mediation`, `WHO_indicator_codebook`, `WHO_mediation_analysis()`

Examples

```
# Global infant-mortality model: GDP growth -> health spending -> IMR,
# with the preprocessing and the grid of the article's Table 1; the screen
# identifies gge_gdp. HBIC chooses 0.1, the smallest value of that grid, so
# the fit warns (class poemed_grid_boundary); WHO_mediation_analysis()
# keeps the article's grid and does not raise the warning.
des <- WHO_mediation_design("imr")
c(n = des$n, mediators = ncol(des$M))
fit <- pe_mediation(scale(des$X), des$Y - mean(des$Y), scale(des$M),
  Z = des$Z, outcome = "continuous", scale = FALSE,
  lambda_grid = seq(0.1, 2, length.out = 100))
fit
# The identified mediator's full name:
WHO_indicator_codebook$description[
  match(des$indicators[attr(fit, "active_mediators")],
        WHO_indicator_codebook$indicator)]

# A region-specific model (Europe).
des_eur <- WHO_mediation_design("imr", region = "EUR")
```

Index

∗ **WHO mediation data**
WHO_health_mediation, 51
WHO_indicator_codebook, 54
WHO_mediation_analysis, 55
WHO_mediation_design, 59
∗ **datasets**
example_data, 3
guo_calibration, 4
WHO_health_mediation, 51
WHO_indicator_codebook, 54
∗ **mediation simulation**
guo_calibration, 4
mediation_ar1_cov, 6
pe_identification_study, 7
pe_power_curve, 32
pe_simulation_study, 36
plot.poemed_tbl, 38
simulate_guo_mediation, 42
simulate_mediation_data, 44
ss_power_pe_mediation, 47
∗ **mediation tests**
pe_mediate, 10
pe_mediation, 12
pe_mediation_linear, 18
pe_mediation_logistic, 23
pe_mediation_poisson, 26
pe_mediators, 30
pe_selection, 35
summary.poemed_tbl, 50
[.poemed_tbl *(poemed_tbl)*, 40

example_binary *(example_data)*, 3
example_continuous *(example_data)*, 3
example_count *(example_data)*, 3
example_data, 3

format, *41*
format.poemed_tbl *(poemed_tbl)*, 40

glance *(poemed_broom)*, 39
guo_calibration, 4, *6*, *9*, *34*, *37*, *39*, *42–44*, *47*, *49*

mediation_ar1_cov, 6
mediation_ar1_cov(), *5*, *9*, *34*, *37*, *39*, *44*, *46*, *47*, *49*

pe_identification_study, 7
pe_identification_study(), *5*, *6*, *34*, *37*, *39*, *44*, *47*, *49*
pe_mediate, 10
pe_mediate(), *17*, *22*, *26*, *30*, *31*, *35*, *51*
pe_mediation, 12
pe_mediation(), *4*, *8*, *10–12*, *18*, *20*, *22*, *23*, *25–27*, *29–31*, *33–35*, *38–40*, *47–49*, *51*, *53*, *56*, *59–61*
pe_mediation_linear, 18
pe_mediation_linear(), *11*, *12*, *14*, *15*, *17*, *20*, *25*, *26*, *28–31*, *35*, *51*, *57*
pe_mediation_logistic, 23
pe_mediation_logistic(), *12*, *14*, *17*, *19*, *22*, *24*, *28*, *30*, *31*, *35*, *51*
pe_mediation_poisson, 26
pe_mediation_poisson(), *12*, *17*, *22*, *26*, *31*, *35*, *51*
pe_mediators, 30
pe_mediators(), *12*, *17*, *21*, *22*, *26*, *30*, *35*, *38–40*, *51*
pe_power_curve, 32
pe_power_curve(), *5–7*, *9*, *17*, *36–39*, *44*, *47*, *49*
pe_selection, 35
pe_selection(), *9*, *12*, *13*, *17*, *19*, *21*, *22*, *24*, *26*, *28*, *30*, *31*, *51*
pe_simulation_study, 36
pe_simulation_study(), *5*, *6*, *9*, *34*, *39*, *44*, *47*, *49*
plot.default(), *38*
plot.poemed_tbl, 38
plot.poemed_tbl(), *5*, *6*, *9*, *34*, *37*, *44*, *47*, *49*

poemed_broom, 39
poemed_tbl, 40
print.data.frame, *40, 41*
print.poemed_tbl *(*poemed_tbl*)*, 40
print.poemed_tbl(), *51*
print.summary.poemed_tbl
*(*summary.poemed_tbl*)*, 50

rbind, *41*
rbind.data.frame, *41*
rbind.poemed_tbl *(*poemed_tbl*)*, 40

simulate_guo_mediation, 42
simulate_guo_mediation(), *4–6*, *9*, *34*, *37*, *39*, *47*, *49*
simulate_mediation_data, 44
simulate_mediation_data(), *3–9*, *17*, *22*, *32–34*, *37*, *39*, *42–44*, *49*
ss_power_pe_mediation, 47
ss_power_pe_mediation(), *5, 6*, *9*, *34*, *37*, *39*, *44*, *47*
subset, *42*
summary(), *58*
summary.data.frame(), *50, 51*
summary.poemed_tbl, 50
summary.poemed_tbl(), *12*, *17*, *22*, *26*, *30, 31*, *35*

tidy *(*poemed_broom*)*, 39

WHO_health_mediation, 51, *54–56*, *59–61*
WHO_indicator_codebook, *52, 53*, 54, *59*, *61*
WHO_mediation_analysis, 55
WHO_mediation_analysis(), *50–53*, *55*, *60, 61*
WHO_mediation_design, 59
WHO_mediation_design(), *12*, *53*, *55, 56*, *59*