Package 'gsaot'

May 8, 2026

Title Compute Global Sensitivity Analysis Indices Using Optimal Transport

Version 1.1.1

Description Computing Global Sensitivity Indices from given data using Optimal Transport, as defined in Borgonovo et al (2024) <doi:10.1287/mnsc.2023.01796>. You provide an input sample, an output sample, decide the algorithm, and compute the indices.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Imports boot, ggplot2, patchwork (>= 1.2.0), Rcpp, RcppEigen (>= 0.3.4.0.0), Rdpack (>= 2.4), stats, transport (>= 0.15.0)

URL https://github.com/pietrocipolla/gsaot, https://pietrocipolla.github.io/gsaot/

BugReports https://github.com/pietrocipolla/gsaot/issues

LinkingTo Rcpp, RcppEigen

VignetteBuilder knitr

RdMacros Rdpack

NeedsCompilation yes

Author Leonardo Chiani [aut, cre, cph] (ORCID: <https://orcid.org/0009-0007-2491-6290>), Emanuele Borgonovo [rev], Elmar Plischke [rev], Massimo Tavoni [rev]

Maintainer Leonardo Chiani <leonardo.chiani@polimi.it>

Repository CRAN

Date/Publication 2025-09-17 17:10:02 UTC

Contents

`confint.gsaot_indices` *Compute confidence intervals for sensitivity indices*

Description

Computes confidence intervals for a `gsaot_indices` object using bootstrap results.

Usage

```
## S3 method for class 'gsaot_indices'
confint(object, parm = NULL, level = 0.95, type = "norm", ...)
```

Arguments

`object`	An object of class `gsaot_indices`, with bootstrap results included.
`parm`	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	(default is 0.95) Confidence level for the interval.
`type`	(default is `"norm"`) Method to compute the confidence interval. For more information, check the `type` option of `boot::boot.ci()`.
`...`	Additional arguments (currently unused).

Value

A data frame with the following columns:

- `input`: Name of the input variable.
- `component`: The index component for Wasserstein-Bures.
- `index`: Estimated indices
- `original`: Original estimates.
- `bias`: Bootstrap bias estimate.
- `low.ci`: Lower bound of the confidence interval.
- `high.ci`: Upper bound of the confidence interval.

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)
y <- y

res <- ot_indices_wb(x, y, 10, boot = TRUE, R = 100)
confint(res, parm = c(1,3), level = 0.9)
```

---

entropic_bound *Entropic lower bounds for entropic optimal transport sensitivity indices*

Description

Calculate entropic lower bounds for entropic Optimal Transport sensitivity indices

Usage

```
entropic_bound(
  y,
  M,
  cost = "L2",
  discrete_out = FALSE,
  solver = "sinkhorn",
  solver_optns = NULL,
  scaling = TRUE
)
```

Arguments

y	An array or a matrix containing the output values.
M	A scalar representing the number of partitions for continuous inputs.
cost	(default "L2") A string or function defining the cost function of the Optimal Transport problem. It should be "L2" or a function taking as input y and returning a cost matrix. If cost="L2", ot_indices uses the squared Euclidean metric.
discrete_out	(default FALSE) Logical, by default the output sample in y are equally weighted. If discrete_out=TRUE, the function tries to create an histogram of the realizations and to use the histogram as weights. It works if the output is discrete or mixed and the number of realizations is large. The advantage of this option is to reduce the dimension of the cost matrix.
solver	Solver for the Optimal Transport problem. Currently supported options are: • "sinkhorn" (default), the Sinkhorn's solver (Cuturi 2013). • "sinkhorn_log", the Sinkhorn's solver in log scale (Peyré et al. 2019).
solver_optns	(optional) A list containing the options for the Optimal Transport solver. See details for allowed options and default ones.
scaling	(default TRUE) Logical that sets whether or not to scale the cost matrix.

Details

The function allows the computation of the entropic lower bounds. solver should be either "sinkhorn" or "sinkhorn_log".

Value

A scalar representing the entropic lower bound.

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)
```

```
x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

M <- 25

sink_lb <- entropic_bound(y, M)
```

`gaussian_fun` *Multivariate Gaussian linear model evaluation*

Description

Generates samples from a multivariate Gaussian distribution and evaluates a simple linear transformation model.

Usage

```
gaussian_fun(N)
```

Arguments

`N`	Number of input samples to generate.

Details

Inputs `x` are sampled from:

$$\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \quad \boldsymbol{\mu} = [1, 1, 1], \quad \Sigma = \begin{bmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{bmatrix}$$

The output is given by:

$$\mathbf{Y} = A\mathbf{X}^\top, \quad A = \begin{bmatrix} 4 & -2 & 1 \\ 2 & 5 & -1 \end{bmatrix}$$

Value

A list with two elements:

- `x`: a numeric matrix of size `N` x `8` containing the input samples.
- `y`: a numeric vector of length `N` with the corresponding function outputs.

See Also

sobol_fun, ishi_homma_fun

Examples

```
result <- gaussian_fun(1000)
head(result$x)
head(result$y)
```

---

irrelevance_threshold *Irrelevance threshold for optimal transport sensitivity indices*

Description

Calculate irrelevance threshold using dummy variable for Optimal Transport sensitivity indices

Usage

```
irrelevance_threshold(
  y,
  M,
  dummy_optns = NULL,
  cost = "L2",
  discrete_out = FALSE,
  solver = "sinkhorn",
  solver_optns = NULL,
  scaling = TRUE
)
```

Arguments

y	An array or a matrix containing the output values.
M	A scalar representing the number of partitions for continuous inputs.
dummy_optns	(default NULL) A list containing the options on the distribution of the dummy variable. See details for more information.
cost	(default "L2") A string or function defining the cost function of the Optimal Transport problem. It should be "L2" or a function taking as input y and returning a cost matrix. If cost="L2", ot_indices uses the squared Euclidean metric.
discrete_out	(default FALSE) Logical, by default the output sample in y are equally weighted. If discrete_out=TRUE, the function tries to create an histogram of the realizations and to use the histogram as weights. It works if the output is discrete or mixed and the number of realizations is large. The advantage of this option is to reduce the dimension of the cost matrix.
solver	Solver for the Optimal Transport problem. Currently supported options are:

- "1d", the one-dimensional analytic solution.
- "wasserstein-bures", the Wasserstein-Bures solution.
- "sinkhorn" (default), the Sinkhorn's solver (Cuturi 2013).
- "sinkhorn_log", the Sinkhorn's solver in log scale (Peyré et al. 2019).
- "transport", a solver of the non regularized OT problem using transport::transport().

solver_optns (optional) A list containing the options for the Optimal Transport solver. See details for allowed options and default ones.

scaling (default TRUE) Logical that sets whether or not to scale the cost matrix.

Details

The function allows the computation of irrelevance threshold. The function samples from a distribution defined in dummy_optns (by default a standard normal), independent from the output y and then computes the indices using the algorithm specified in solver. Under the hood, lower_bound calls the other available functions in the package:

- ot_indices_1d() (for solver="1d")
- ot_indices_wb() (for solver="wasserstein-bures")
- ot_indices() (for solver %in% c("sinkhorn", "sinkhorn_log", "wasserstein")) The user can choose the distribution of the dummy variable using the argument dummy_optns. dummy_optns should be a named list with at least a term called "distr" defining the sampling function. The other terms in the list are used as arguments to the sampling function.

Value

An object of class gsaot_indices.

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

M <- 25

dummy_lb <- irrelevance_threshold(y, M)

# Custom sampling funtion and network simplex solver
```

```
dummy_optns <- list(distr = "rgamma", shape = 3)
dummy_lb_cust <- irrelevance_threshold(y, M,
                                       dummy_optns = dummy_optns,
                                       solver = "transport")
```

`ishi_homma_fun` *Ishigami-Homma function evaluation*

Description

Evaluates the Ishigami-Homma function. Input samples are drawn from a uniform distribution over $[-\pi, \pi]^3$

Usage

```
ishi_homma_fun(N, A = 2, B = 1)
```

Arguments

`N`	Number of input samples to generate.
`A`	(default: `2`) Numeric, amplitude of the second sine component .
`B`	(default: `1`) Numeric, coefficient of the interaction term.

Details

The Ishigami-Homma function is defined as:

$$Y = \sin(X_1) + A \cdot \sin^2(X_2) + B \cdot X_3^4 \cdot \sin(X_1)$$

where $X_i \sim \mathcal{U}(-\pi, \pi)$.

Value

A list with two elements:

- `x`: a numeric matrix of size `N` x `8` containing the input samples.
- `y`: a numeric vector of length `N` with the corresponding function outputs.

See Also

`sobol_fun`, `gaussian_fun`

Examples

```
result <- ishi_homma_fun(1000)
head(result$x)
head(result$y)
```

`ot_indices` *Estimate optimal transport sensitivity indices for generic outputs*

Description

`ot_indices` calculates sensitivity indices using Optimal Transport (OT) for a multivariate output sample `y` with respect to input data `x`. Sensitivity indices measure the influence of inputs on outputs, with values ranging between 0 and 1.

Usage

```
ot_indices(
  x,
  y,
  M,
  cost = "L2",
  discrete_out = FALSE,
  solver = "sinkhorn",
  solver_optns = NULL,
  scaling = TRUE,
  boot = FALSE,
  stratified_boot = TRUE,
  R = NULL,
  parallel = "no",
  ncpus = 1,
  conf = 0.95,
  type = "norm"
)
```

Arguments

`x`	A matrix or data.frame containing the input(s) values. The values can be numeric, factors, or strings. The type of data changes the partitioning. If the values are continuous (double), the function partitions the data into `M` sets. If the values are discrete (integers, strings, factors), the number of partitioning sets is data-driven.
`y`	A matrix containing the output values. Each column represents a different output variable, and each row represents a different observation. Only numeric values are allowed.
`M`	A scalar representing the number of partitions for continuous inputs.
`cost`	(default `"L2"`) A string or function defining the cost function of the Optimal Transport problem. It should be "L2" or a function taking as input y and returning a cost matrix. If `cost="L2"`, `ot_indices` uses the squared Euclidean metric.

`discrete_out` (default `FALSE`) Logical, by default the output sample in `y` are equally weighted. If `discrete_out=TRUE`, the function tries to create an histogram of the realizations and to use the histogram as weights. It works if the output is discrete or mixed and the number of realizations is large. The advantage of this option is to reduce the dimension of the cost matrix.

`solver` Solver for the Optimal Transport problem. Currently supported options are:

- `"sinkhorn"` (default), the Sinkhorn's solver (Cuturi 2013).
- `"sinkhorn_log"`, the Sinkhorn's solver in log scale (Peyré et al. 2019).
- `"transport"`, a solver of the non regularized OT problem using `transport::transport()`.

`solver_optns` (optional) A list containing the options for the Optimal Transport solver. See details for allowed options and default ones.

`scaling` (default `TRUE`) Logical that sets whether or not to scale the cost matrix.

`boot` (default `FALSE`) Logical that sets whether or not to perform bootstrapping of the OT indices.

`stratified_boot` (default `TRUE`) Logical that sets the type of resampling performed. With `stratified_boot=FALSE`, the function resamples the dataset and then creates the partitions. Otherwise, first, it creates the partitions and then it performs stratified bootstrapping with strata being the partitions.

`R` (default `NULL`) Positive integer, number of bootstrap replicas.

`parallel` (default `"no"`) The type of parallel operation to be used (if any). If missing, the default is taken from the option `boot.parallel` (and if that is not set, `"no"`). Only considered if `boot = TRUE`. For more information, check the `boot::boot()` function.

`ncpus` (default `1`) Positive integer: number of processes to be used in parallel operation: typically one would chose this to the number of available CPUs. Check the `ncpus` option in the `boot::boot()` function of the boot package.

`conf` (default `0.95`) Number between `0` and `1` representing the default confidence level. Only considered if `boot = TRUE`. Different confidence levels can be computed as a postprocessing using `confint.gsaot_indices()`.

`type` (default `"norm"`) Method to compute the default confidence interval. Only considered if `boot = TRUE`. For more information, check the `type` argument of `boot::boot.ci()`. Different confidence intervals can be computed as a postprocessing using `confint.gsaot_indices()`.

Details

Solvers:

OT is a widely studied topic in Operational Research and Calculus. The reference for the OT solvers in this package is Peyré et al. (2019). The default solver is `"sinkhorn"`, the Sinkhorn's solver introduced in Cuturi (2013). It solves the entropic-regularized version of the OT problem. The `"sinkhorn_log"` solves the same OT problem but in log scale. It is more stable for low values of the regularization parameter but slower to converge. The option `"transport"` is used to choose a solver for the non-regularized OT problem. Under the hood, the function calls `transport::transport()` from package `transport`. This option does not define the solver per se, but the solver should be defined with the argument `solver_optns`. See the next section for more information.

Solver options:

The argument `solver_optns` should be empty (for default options) or a list with all or some of the required solver parameters. All the parameters not included in the list will be set to default values. The solvers `"sinkhorn"` and `"sinkhorn_log"` have the same options:

- `numIterations` (default `1e3`): a positive integer defining the maximum number of Sinkhorn's iterations allowed. If the solver does not converge in the number of iterations set, the solver will throw an error.
- `epsilon` (default `0.01`): a positive real number defining the regularization coefficient. If the value is too low, the solver may return `NA`.
- `maxErr` (default `1e-9`): a positive real number defining the approximation error threshold between the marginal histogram of the partition and the one computed by the solver. The solver may fail to converge in `numIterations` if this value is too low.

The solver `"transport"` has the parameters:

- `method` (default `"networkflow`): string defining the solver of the OT problem.
- `control`: a named list of parameters for the chosen method or the result of a call to `transport::trcontrol()`.
- `threads` (default `1`): an Integer specifying the number of threads used in parallel computing.

For details regarding this solver, check the `transport::transport()` help page.

Value

A `gsaot_indices` object containing:

- `method`: a string that identifies the type of indices computed.
- `indices`: a names array containing the sensitivity indices between 0 and 1 for each column in `x`, indicating the influence of each input variable on the output variables.
- `bound`: a double representing the upper bound of the separation measure or an array representing the mean of the separation for each input according to the bootstrap replicas.
- `x`, `y`: input and output data provided as arguments of the function.
- `inner_statistic`: a list of matrices containing the values of the inner statistics for the partitions defined by `partitions`. If `method = wasserstein-bures`, each matrix has three rows containing the Wasserstein-Bures indices, the Advective, and the Diffusive components.
- `partitions`: a matrix containing the partitions built to calculate the sensitivity indices. Each column contains the partition associated to the same column in `x`.

If `boot = TRUE`, the object contains also:

- `indices_ci`: a `data.frame` with first column the input, second and third columns the lower and upper bound of the confidence interval.
- `inner_statistic_ci`: a list of matrices. Each element of the list contains the lower and upper confidence bounds for the partition defined by the row.
- `bound_ci`: a list containing the lower and upper bounds of the confidence intervals of the separation measure bound.
- `type`, `conf`: type of confidence interval and confidence level, provided as arguments.
- `W_boot`: list of bootstrap objects, one for each input.

References

Cuturi M (2013). "Sinkhorn distances: Lightspeed computation of optimal transport." *Advances in neural information processing systems*, **26**.

Peyré G, Cuturi M, others (2019). "Computational optimal transport: With applications to data science." *Foundations and Trends® in Machine Learning*, **11**(5-6), 355–607.

See Also

ot_indices_1d(), ot_indices_wb()

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)

M <- 25

# Calculate sensitivity indices
sensitivity_indices <- ot_indices(x, y, M)
sensitivity_indices
```

---

ot_indices_1d *Estimate optimal transport indices on one dimensional outputs*

Description

Estimate optimal transport indices on one dimensional outputs

Usage

```
ot_indices_1d(
  x,
  y,
```

```
  M,
  p = 2,
  boot = FALSE,
  R = NULL,
  parallel = "no",
  ncpus = 1,
  conf = 0.95,
  type = "norm"
)
```

Arguments

x	A matrix or data.frame containing the input(s) values. The values can be numeric, factors, or strings. The type of data changes the partitioning. If the values are continuous (double), the function partitions the data into M sets. If the values are discrete (integers, strings, factors), the number of partitioning sets is data-driven.
y	An array containing the output values.
M	A scalar representing the number of partitions for continuous inputs.
p	A numeric representing the p-norm Lp used as ground cost in the Optimal Transport problem.
boot	(default FALSE) Logical that sets whether or not to perform bootstrapping of the OT indices.
R	(default NULL) Positive integer, number of bootstrap replicas.
parallel	(default "no") The type of parallel operation to be used (if any). If missing, the default is taken from the option boot.parallel (and if that is not set, "no"). Only considered if boot = TRUE. For more information, check the boot::boot() function.
ncpus	(default 1) Positive integer: number of processes to be used in parallel operation: typically one would chose this to the number of available CPUs. Check the ncpus option in the boot::boot() function of the boot package.
conf	(default 0.95) Number between 0 and 1 representing the default confidence level. Only considered if boot = TRUE. Different confidence levels can be computed as a postprocessing using confint.gsaot_indices().
type	(default "norm") Method to compute the default confidence interval. Only considered if boot = TRUE. For more information, check the type argument of boot::boot.ci(). Different confidence intervals can be computed as a postprocessing using confint.gsaot_indices().

Value

A gsaot_indices object containing:

- method: a string that identifies the type of indices computed.
- indices: a names array containing the sensitivity indices between 0 and 1 for each column in x, indicating the influence of each input variable on the output variables.

- bound: a double representing the upper bound of the separation measure or an array representing the mean of the separation for each input according to the bootstrap replicas.
- x, y: input and output data provided as arguments of the function.
- inner_statistic: a list of matrices containing the values of the inner statistics for the partitions defined by partitions. If method = wasserstein-bures, each matrix has three rows containing the Wasserstein-Bures indices, the Advective, and the Diffusive components.
- partitions: a matrix containing the partitions built to calculate the sensitivity indices. Each column contains the partition associated to the same column in x.

If boot = TRUE, the object contains also:

- indices_ci: a data.frame with first column the input, second and third columns the lower and upper bound of the confidence interval.
- inner_statistic_ci: a list of matrices. Each element of the list contains the lower and upper confidence bounds for the partition defined by the row.
- bound_ci: a list containing the lower and upper bounds of the confidence intervals of the separation measure bound.
- type, conf: type of confidence interval and confidence level, provided as arguments.
- W_boot: list of bootstrap objects, one for each input.

See Also

ot_indices(), ot_indices_wb()

Examples

```
x <- rnorm(1000)
y <- 10 * x
ot_indices_1d(data.frame(x), y, 10)
```

---

ot_indices_smap *Estimate sensitivity maps using optimal transport indices*

Description

Estimate sensitivity maps using optimal transport indices

Usage

```
ot_indices_smap(x, y, M)
```

Arguments

x	A matrix or data.frame containing the input(s) values. The values can be numeric, factors, or strings. The type of data changes the partitioning. If the values are continuous (double), the function partitions the data into M sets. If the values are discrete (integers, strings, factors), the number of partitioning sets is data-driven.
y	A matrix containing the output values. Each column is interpreted as a different output.
M	A scalar representing the number of partitions for continuous inputs.

Value

A matrix where each column represents an input and each row represents an output. The values are indices between 0 and 1 computed using `ot_indices_1d()`.

Examples

```
N <- 1000

x1 <- rnorm(N)
x2 <- rnorm(N)
x <- cbind(x1, x2)

y1 <- 10 * x1
y2 <- x1 + x2
y <- cbind(y1, y2)

ot_indices_smap(data.frame(x), y, 30)
```

---

ot_indices_wb *Estimate Wasserstein-Bures approximation of the optimal transport solution*

Description

Estimate Wasserstein-Bures approximation of the optimal transport solution

Usage

```
ot_indices_wb(
  x,
  y,
  M,
  boot = FALSE,
  R = NULL,
  parallel = "no",
  ncpus = 1,
```

```
  conf = 0.95,
  type = "norm"
)
```

Arguments

x	A matrix or data.frame containing the input(s) values. The values can be numeric, factors, or strings. The type of data changes the partitioning. If the values are continuous (double), the function partitions the data into M sets. If the values are discrete (integers, strings, factors), the number of partitioning sets is data-driven.
y	A matrix containing the output values. Each column represents a different output variable, and each row represents a different observation. Only numeric values are allowed.
M	A scalar representing the number of partitions for continuous inputs.
boot	(default FALSE) Logical that sets whether or not to perform bootstrapping of the OT indices.
R	(default NULL) Positive integer, number of bootstrap replicas.
parallel	(default "no") The type of parallel operation to be used (if any). If missing, the default is taken from the option boot.parallel (and if that is not set, "no"). Only considered if boot = TRUE. For more information, check the boot::boot() function.
ncpus	(default 1) Positive integer: number of processes to be used in parallel operation: typically one would chose this to the number of available CPUs. Check the ncpus option in the boot::boot() function of the boot package.
conf	(default 0.95) Number between 0 and 1 representing the default confidence level. Only considered if boot = TRUE. Different confidence levels can be computed as a postprocessing using confint.gsaot_indices().
type	(default "norm") Method to compute the default confidence interval. Only considered if boot = TRUE. For more information, check the type argument of boot::boot.ci(). Different confidence intervals can be computed as a postprocessing using confint.gsaot_indices().

Value

A gsaot_indices object containing:

- method: a string that identifies the type of indices computed.
- indices: a names array containing the sensitivity indices between 0 and 1 for each column in x, indicating the influence of each input variable on the output variables.
- bound: a double representing the upper bound of the separation measure or an array representing the mean of the separation for each input according to the bootstrap replicas.
- x, y: input and output data provided as arguments of the function.
- inner_statistic: a list of matrices containing the values of the inner statistics for the partitions defined by partitions. If method = wasserstein-bures, each matrix has three rows containing the Wasserstein-Bures indices, the Advective, and the Diffusive components.

- `partitions`: a matrix containing the partitions built to calculate the sensitivity indices. Each column contains the partition associated to the same column in `x`.

If `boot = TRUE`, the object contains also:

- `indices_ci`: a `data.frame` with first column the input, second and third columns the lower and upper bound of the confidence interval.
- `inner_statistic_ci`: a list of matrices. Each element of the list contains the lower and upper confidence bounds for the partition defined by the row.
- `bound_ci`: a list containing the lower and upper bounds of the confidence intervals of the separation measure bound.
- `type`, `conf`: type of confidence interval and confidence level, provided as arguments.
- `W_boot`: list of bootstrap objects, one for each input.

See Also

`ot_indices()`, `ot_indices_1d()`

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)
y <- y

ot_indices_wb(x, y, 10)
```

---

plot.gsaot_indices *Plot optimal transport sensitivity indices*

Description

Plot Optimal Transport based sensitivity indices using `ggplot2` package.

Usage

```
## S3 method for class 'gsaot_indices'
plot(x, ranking = NULL, wb_all = FALSE, threshold = NULL, ...)
```

Arguments

x	An object generated by ot_indices, ot_indices_1d, or ot_indices_wb.
ranking	An integer with absolute value less or equal than the number of inputs. If positive, select the first ranking inputs per importance. If negative, select the last ranking inputs per importance.
wb_all	(default FALSE) Logical that defines whether or not to plot the Advective and Diffusive components of the Wasserstein-Bures indices.
threshold	(default NULL) A double or and object of class gsaot_indices that represents a lower threshold.
...	Further arguments passed to or from other methods.

Value

A ggplot object that, if called, will print

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)

M <- 25

# Calculate sensitivity indices
sensitivity_indices <- ot_indices_wb(x, y, M)
sensitivity_indices

plot(sensitivity_indices)
```

plot_comparison *Compare sensitivity indices across methods*

Description

This function takes a list of gsaot_indices objects and generates a bar plot comparing the sensitivity indices across different methods.

Usage

```
plot_comparison(x_list, wb_all = FALSE)
```

Arguments

x_list	A list of S3 objects of class "gsaot_indices", each representing sensitivity analysis results for a different solver.
wb_all	(default FALSE) Logical that defines whether or not to plot the Advective and Diffusive components of the Wasserstein-Bures indices.

Value

A ggplot object representing the bar plot of sensitivity indices grouped by input and colored by method.

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)

M <- 25

# Calculate sensitivity indices
ind_wb <- ot_indices_wb(x, y, M)
ind_sink <- ot_indices(x, y, M)
```

```
plot_comparison(list(ind_wb, ind_sink))
```

`plot_separations` *Plot optimal transport local separations*

Description

Plot Optimal Transport based local separations for each partition using `ggplot2` package. If provided, it plots also the uncertainty estimates.

Usage

```
plot_separations(x, ranking = NULL, wb_all = FALSE, ...)
```

Arguments

`x`	An object generated by `ot_indices`, `ot_indices_1d`, or `ot_indices_wb`.
`ranking`	An integer with absolute value less or equal than the number of inputs. If positive, select the first `ranking` inputs per importance. If negative, select the last `ranking` inputs per importance.
`wb_all`	(default `FALSE`) Logical that defines whether or not to plot the Advective and Diffusive components of the Wasserstein-Bures indices.
`...`	Further arguments passed to or from other methods.

Value

A `patchwork` object that, if called, will print.

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)
```

```
M <- 25

# Get sensitivity indices
sensitivity_indices <- ot_indices(x, y, M)
plot_separations(sensitivity_indices)
```

`print.gsaot_indices` *Print optimal transport sensitivity indices information*

Description

Print optimal transport sensitivity indices information

Usage

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

Arguments

`x`	An object generated by `ot_indices`, `ot_indices_1d`, or `ot_indices_wb`.
`...`	Further arguments passed to or from other methods.

Value

The information contained in argument `x`

Examples

```
N <- 1000

mx <- c(1, 1, 1)
Sigmax <- matrix(data = c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)

x1 <- rnorm(N)
x2 <- rnorm(N)
x3 <- rnorm(N)

x <- cbind(x1, x2, x3)
x <- mx + x %*% chol(Sigmax)

A <- matrix(data = c(4, -2, 1, 2, 5, -1), nrow = 2, byrow = TRUE)
y <- t(A %*% t(x))

x <- data.frame(x)

M <- 25
```

```
# Calculate sensitivity indices
sensitivity_indices <- ot_indices(x, y, M)
print(sensitivity_indices)
```

`sobol_fun` *Sobol G function evaluation*

Description

This function evaluates the Sobol G function on a set of input samples generated via crude Monte Carlos. It returns both the sampled inputs and the corresponding function outputs.

Usage

```
sobol_fun(N, a = c(0, 1, 4.5, 9, 99, 99, 99, 99))
```

Arguments

`N`	Integer. Number of input samples to generate.
`a`	(default: `c(0, 1, 4.5, 9, 99, 99, 99, 99)`) Numeric vector of non-negative parameters of length 8. These parameters control the sensitivity of each input dimension.

Details

The Sobol G function is defined as:

$$Y = \prod_{j=1}^{8} \frac{| 4X_j - 2 | + a_j}{1 + a_j}$$

where $X_j \sim \mathcal{U}(0, 1)$ independently.

Value

A list with two elements:

- `x`: a numeric matrix of size `N` x `8` containing the input samples.
- `y`: a numeric vector of length `N` with the corresponding function outputs.

See Also

`ishi_homma_fun`, `gaussian_fun`

Examples

```
result <- sobol_fun(1000)
head(result$x)
head(result$y)
```

summary.gsaot_indices *Summary method for* gsaot_indices *objects*

Description

Summary method for gsaot_indices objects

Usage

```
## S3 method for class 'gsaot_indices'
summary(object, digits = 3, ranking = NULL, ...)
```

Arguments

object	An object of class "gsaot_indices".
digits	(default: 3) Number of significant digits to print for numeric values.
ranking	An integer with absolute value less or equal than the number of inputs. If positive, select the first ranking inputs per importance.
...	Further arguments (currently ignored).

Value

(Invisibly) a named list containing the main elements summarised on screen.

Index