# Package: infoxtr (via r-universe)

May 26, 2026

**Title** Information-Theoretic Measures for Revealing Variable Interactions

**Version** 0.3

**Description** Implements information-theoretic measures to explore variable interactions, including KSG mutual information estimation for continuous variables from Kraskov et al. (2004) <doi:10.1103/PhysRevE.69.066138>, knockoff conditional mutual information described in Zhang & Chen (2025) <doi:10.1126/sciadv.adu6464>, synergistic-unique-redundant decomposition introduced by Martinez-Sanchez et al. (2024) <doi:10.1038/s41467-024-53373-4>, allowing detection of complex and diverse relationships among variables.

**License** GPL-3

**Encoding** UTF-8

**URL** https://stscl.github.io/infoxtr/, https://github.com/stscl/infoxtr

**BugReports** https://github.com/stscl/infoxtr/issues

**Depends** R (>= 4.1.0)

**LinkingTo** Rcpp, RcppThread

**Imports** methods, sdsfun, sf, terra

**Suggests** knitr, Rcpp, RcppThread, readr, rmarkdown, spEDM, tEDM

**VignetteBuilder** knitr

**Config/roxygen2/markdown** TRUE

**Config/roxygen2/version** 8.0.0

**Config/pak/sysreqs** libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

**Repository** https://stscl.r-universe.dev

**Date/Publication** 2026-05-26 08:36:00 UTC

**RemoteUrl** https://github.com/stscl/infoxtr

**RemoteRef** HEAD

**RemoteSha** 22cb4a9edc479616d9b7592cb2f4f4258df2b42b

# Contents



---

ce *Conditional Entropy*

---

## Description

Estimate the conditional entropy of target variables given conditioning variables.

## Usage

```
ce(data, target, conds, base = exp(1), type = c("cont", "disc"), k = 3)
```

## Arguments

| | |
|---|---|
| `data` | Observation data. |
| `target` | Integer vector of column indices for the target variables. |
| `conds` | Integer vector of column indices for the conditioning variables. |
| `base` | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| `type` | (optional) Estimation method: `"disc"` for discrete entropy or `"cont"` for continuous entropy (KSG estimator). |
| `k` | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when `type = "disc"`. |

## Value

A numerical value.

## Examples

```
infoxtr::ce(matrix(1:100,ncol=2),1,2)
```

cmi *Conditional Mutual Information*

## Description

Estimate the conditional mutual information between target and interacting variables given conditioning variables.

## Usage

```
cmi(
  data,
  target,
  interact,
  conds,
  base = exp(1),
  type = c("cont", "disc"),
  k = 3,
  normalize = FALSE
)
```

## Arguments

| | |
|---|---|
| data | Observation data. |
| target | Integer vector of column indices for the target variables. |
| interact | Integer vector of column indices for the interacting variables. |
| conds | Integer vector of column indices for the conditioning variables. |
| base | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| type | (optional) Estimation method: `"disc"` for discrete entropy or `"cont"` for continuous entropy (KSG estimator). |
| k | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when `type = "disc"`. |
| normalize | (optional) Logical; if `TRUE`, return normalized mutual information. |

## Value

A numerical value.

## Examples

```
set.seed(42)
infoxtr::cmi(matrix(stats::rnorm(99,1,10),ncol=3),1,2,3)
```

`discretize` *Discretization*

## Description

Discretize a numeric vector into categorical classes using several commonly used discretization methods. Missing values (`NA`/`NaN`) are ignored and returned as class `0`.

## Usage

```
discretize(
  x,
  n = 5,
  method = "natural",
  large = 3000,
  prop = 0.15,
  seed = 42,
  thr = 0.4,
  iter = 100,
  bps = NULL,
  right_closed = TRUE
)
```

## Arguments

| | |
|---|---|
| `x` | A vector. |
| `n` | (optional) Number of classes. |
| `method` | (optional) Discretization method. One of `"sd"`, `"equal"`, `"geometric"`, `"quantile"`, `"manual"`, `"natural("jenks")"`, or `"headtail"("headtails")`. |
| `large` | (optional) Threshold sample size for natural breaks sampling. |
| `prop` | (optional) Sampling proportion used when `method = "natural"` and the input size exceeds `large`. |
| `seed` | (optional) Random seed used for sampling in natural breaks. |
| `thr` | (optional) Threshold used in the head/tail breaks algorithm. |
| `iter` | (optional) Maximum number of iterations for head/tail breaks. |
| `bps` | (optional) Numeric vector of manual breakpoints used when `method = "manual"`. |
| `right_closed` | (optional) Logical. If `TRUE`, intervals are right-closed (e.g., `(a, b]`). If `FALSE`, intervals are left-closed `[a, b)`. |

## Value

A discretized integer vector.

## Note

If `x` is not numeric, it will be converted to integer categories via `as.factor()`.

## Examples

```
set.seed(42)
infoxtr::discretize(stats::rnorm(99,1,10))
```

---

`entropy` *Shannon Entropy*

---

## Description

Estimate the entropy of a vector using either category counts (for discrete data) or a k-nearest neighbor estimator (for continuous data).

## Usage

```
entropy(vec, base = exp(1), type = c("cont", "disc"), k = 3)
```

## Arguments

| | |
|---|---|
| `vec` | A vector. |
| `base` | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| `type` | (optional) Estimation method: `"disc"` for discrete entropy or `"cont"` for continuous entropy (KSG estimator). |
| `k` | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when `type = "disc"`. |

## Value

A numerical value.

## Examples

```
set.seed(42)
infoxtr::entropy(stats::rnorm(100), type = "cont")
infoxtr::entropy(sample(letters[1:5], 100, TRUE), base = 2, type = "disc")
```

## je *Joint Entropy*

### Description

Estimate the joint entropy of selected variables.

### Usage

```
je(data, indices, base = exp(1), type = c("cont", "disc"), k = 3)
```

### Arguments

| | |
|---|---|
| data | Observation data. |
| indices | Integer vector of column indices to include in joint entropy calculation. |
| base | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| type | (optional) Estimation method: `"disc"` for discrete entropy or `"cont"` for continuous entropy (KSG estimator). |
| k | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when `type = "disc"`. |

### Value

A numerical value.

### Examples

```
infoxtr::je(matrix(1:100,ncol=2),1:2)
```

## kocmi *KOCMI*

### Description

Knockoff Conditional Mutual Information

## Usage

```
kocmi(
  data,
  target,
  agent,
  conds,
  knockoff,
  null_knockoff = NULL,
  type = c("cont", "disc"),
  nboots = 10000,
  k = 3,
  threads = 1,
  seed = 42,
  base = exp(1),
  method = "equal",
  contain_null = TRUE
)
```

## Arguments

| | |
|---|---|
| `data` | Observation data. |
| `target` | Integer vector of column indices for the target variables. |
| `agent` | Integer vector of column indices for the source (agent) variables. |
| `conds` | Integer vector of column indices for the conditioning variables. |
| `knockoff` | Knockoff realizations constructed for the `agent` variable while keeping the `target` variable unchanged. Each column corresponds to one Monte Carlo knockoff sample generated using the remaining variables except the target. |
| `null_knockoff` | (optional) Knockoff realizations generated under the null setting where all variables are jointly used to construct knockoffs. Each column represents one Monte Carlo sample. If `contain_null = FALSE`, this argument can be `NULL`. |
| `type` | (optional) Estimation method: `"disc"` for discrete mutual information or `"cont"` for continuous mutual information (KSG estimator). |
| `nboots` | (optional) Number of permutations used in the sign-flipping permutation test for evaluating the significance of the mean information difference. |
| `k` | (optional) For `type = "cont"`, the number of nearest neighbors used by the continuous conditional mutual information estimator. For `type = "disc"`, the number of bins used for discretization. |
| `threads` | (optional) Number of threads used. |
| `seed` | (optional) Random seed used for permutation test. |
| `base` | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| `method` | (optional) Discretization method. One of `"sd"`, `"equal"`, `"geometric"`, `"quantile"`, `"natural("jenks")"`, or `"headtail"("headtails")`. |

`contain_null` (optional) Logical. If `TRUE`, the test statistic is computed using knockoffs generated under the null model (provided in `null_knockoff`). In this case the difference is defined as $I(Y; X_{null}|Z) - I(Y; X_{knockoff}|Z)$. If `FALSE`, the original conditional mutual information $I(Y; X|Z)$ is used instead and compared against the knockoff estimates $I(Y; X_{knockoff}|Z)$.

## Value

A named numeric vector.

## Note

`kocmi` only support numeric data.

## References

Zhang, X., Chen, L., 2025. Quantifying interventional causality by knockoff operation. Science Advances 11.

## Examples

```
set.seed(42)
kn1 = replicate(50, stats::rnorm(100))
kn2 = replicate(50, stats::rnorm(100))
mat = replicate(3, stats::rnorm(100))
infoxtr::kocmi(mat, 1, 2, 3, kn1, kn2)
```

---

# mi *Mutual Information*

---

## Description

Estimate the mutual information between target and interacting variables.

## Usage

```
mi(
  data,
  target,
  interact,
  base = exp(1),
  type = c("cont", "disc"),
  k = 3,
  normalize = FALSE
)
```

### Arguments

| | |
|---|---|
| data | Observation data. |
| target | Integer vector of column indices for the target variables. |
| interact | Integer vector of column indices for the interacting variables. |
| base | (optional) Logarithm base of the entropy. Defaults to exp(1) (nats). Use 2 for bits or 10 for dits. |
| type | (optional) Estimation method: "disc" for discrete entropy or "cont" for continuous entropy (KSG estimator). |
| k | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when type = "disc". |
| normalize | (optional) Logical; if TRUE, return normalized mutual information. |

### Value

A numerical value.

### Examples

```
infoxtr::mi(matrix(1:100,ncol=2),1,2)
```

---

## surd *SURD*

---

### Description

Synergistic-Unique-Redundant Decomposition

### Usage

```
## S4 method for signature 'data.frame'
surd(
  data,
  target,
  agent,
  lag = 1,
  bin = 5,
  method = "equal",
  max.order = 10,
  threads = 1,
  base = 2,
  normalize = TRUE
)

## S4 method for signature 'sf'
```

```
surd(
  data,
  target,
  agent,
  lag = 1,
  bin = 5,
  method = "equal",
  max.order = 10,
  threads = 1,
  base = 2,
  normalize = TRUE,
  nb = NULL
)

## S4 method for signature 'SpatRaster'
surd(
  data,
  target,
  agent,
  lag = 1,
  bin = 5,
  method = "equal",
  max.order = 10,
  threads = 1,
  base = 2,
  normalize = TRUE
)
```

## Arguments

| | |
|---|---|
| `data` | Observation data. |
| `target` | Integer vector of column indices for the target variables. |
| `agent` | Integer vector of column indices for the source (agent) variables. |
| `lag` | (optional) Lag of the agent variables. |
| `bin` | (optional) Number of discretization bins. |
| `method` | (optional) Discretization method. One of `"sd"`, `"equal"`, `"geometric"`, `"quantile"`, `"natural("jenks")"`, or `"headtail"("headtails")`. |
| `max.order` | (optional) Maximum combination order. |
| `threads` | (optional) Number of threads used. |
| `base` | (optional) Logarithm base of the entropy. Defaults to `exp(1)` (nats). Use `2` for bits or `10` for dits. |
| `normalize` | (optional) Logical; if `TRUE`, return normalized mutual information. |
| `nb` | (optional) Neighbours list. |

## Value

A list.

**vars** Character vector indicating the variable combination associated with each information component.

**types** Character vector indicating the information type of each component.

**values** Numeric vector giving the magnitude of each information component.

## Note

`surd` only supports numeric input data. Both `bin` and `method` support variable-specific settings using R-style recycling:

- length 1: applied to the target and all agent variables
- length 2: first for the target, second for all agents
- length > 2: first for the target, remaining values are recycled across agents

## References

Martinez-Sanchez, A., Arranz, G., Lozano-Duran, A., 2024. Decomposing causality into its synergistic, unique, and redundant components. Nature Communications 15.

## Examples

```
columbus = sf::read_sf(system.file("case/columbus.gpkg", package="spEDM"))
infoxtr::surd(columbus, 1, 2:3)
```

---

`te` *Transfer Entropy*

---

## Description

Estimate the transfer entropy from agent variables to target variables.

## Usage

```
te(
  data,
  target,
  agent,
  lag_p = 3,
  lag_q = 3,
  base = exp(1),
  type = c("cont", "disc"),
  k = 3,
  normalize = FALSE,
  lag_single = FALSE
)
```

## Arguments

| | |
|---|---|
| data | Observation data. |
| target | Integer vector of column indices for the target variables. |
| agent | Integer vector of column indices for the source (agent) variables. |
| lag_p | (optional) Lag of the target variables. |
| lag_q | (optional) Lag of the agent variables. |
| base | (optional) Logarithm base of the entropy. Defaults to exp(1) (nats). Use 2 for bits or 10 for dits. |
| type | (optional) Estimation method: "disc" for discrete entropy or "cont" for continuous entropy (KSG estimator). |
| k | (optional) Number of nearest neighbors used by the continuous estimator. Ignored when type = "disc". |
| normalize | (optional) Logical; if TRUE, return normalized mutual information. |
| lag_single | (optional) Logical; if FALSE, use full lag embedding. |

## Value

A numerical value.

## References

Schreiber, T., 2000. Measuring Information Transfer. Physical Review Letters 85, 461–464.

## Examples

```
set.seed(42)
infoxtr::te(matrix(stats::rnorm(100,1,10),ncol=2),1,2)
```

# Index