Package: simOutrank (via r-universe)

July 19, 2026

Title Outranking-Based Trace Clustering

Version 0.3.0

Description Implements outranking-based trace clustering for process mining. Pairwise similarity between traces is assessed on multiple, problem-specific criteria using ELECTRE-III-style partial concordance and discordance indices with indifference, similarity and veto thresholds. The aggregated credibility matrix is then clustered using normalized spectral clustering or hierarchical clustering. Also includes outlier trimming, must-link/cannot-link adjustments, validation indices and diagnostic plots.

License GPL-3

Encoding UTF-8

Language en-GB

RoxygenNote 8.0.0

Imports graphics, lpSolve, stats, stringdist

Suggests bupaR, clValid, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 3.5)

LazyData true

VignetteBuilder knitr

URL https://github.com/pdelias/simOutrank, https://pdelias.github.io/simOutrank/

BugReports https://github.com/pdelias/simOutrank/issues

NeedsCompilation no

Author Pavlos Delias [aut, cre] (ORCID: <https://orcid.org/0000-0002-3722-2307>)

Maintainer Pavlos Delias <pavlos.delias@gmail.com>

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

Date/Publication 2026-07-19 13:00:02 UTC

RemoteUrl https://github.com/cran/simOutrank

RemoteRef HEAD

RemoteSha 145c1133c1374878d2aa7d1f3eaedcdc3d72f081

Contents

`as_quantile` *Quantile threshold specification*

Description

`as_quantile()` marks a criterion threshold as a quantile of the empirical distribution of that criterion's off-diagonal pairwise values, rather than a fixed number. It is resolved to a numeric value when the criterion's measure matrix is available (see `criterion()`).

Usage

```
as_quantile(p)
```

Arguments

`p`	A single probability in `[0, 1]`.

Value

An object of class `outrank_quantile`.

Examples

```
as_quantile(0.8)
```

`as_traces` *Represent an event log as traces*

Description

`as_traces()` converts an event log into a `traces` object: the time-ordered activity sequence of every case together with the case-level attribute table. Activities are encoded as integers internally (never as single letters) so that logs with more than 26 activities or with multi-character labels are handled correctly; the original labels are kept for display and decoding.

Usage

```
as_traces(x, ...)

## S3 method for class 'data.frame'
as_traces(x, case_id, activity, timestamp, ...)

## S3 method for class 'eventlog'
as_traces(x, ...)

## S3 method for class 'activitylog'
as_traces(x, ...)
```

Arguments

`x`	An event log. A `data.frame` in which each row is an event, or – when the **bupaR** package is installed – a bupaR `eventlog` or `activitylog`.
`...`	Passed on to methods.
`case_id`, `activity`, `timestamp`	Length-one character strings naming the columns that hold the case identifier, the activity label and the event timestamp. For the bupaR methods the mapping is read from the object and these arguments must not be supplied.

Details

Events are ordered by `timestamp` within each case. Ties (two events of the same case sharing a timestamp) are broken by the original row order and raise a warning. Cases keep their order of first appearance in `x`.

Every column other than `case_id`, `activity` and `timestamp` is treated as a case-level attribute and summarised by its first value within the ordered case, matching the convention that such attributes are constant per case.

Value

An object of class `traces`, a list with elements:

`case_ids` character vector of case identifiers, in order.

`sequences` named list of integer vectors; each entry is the time-ordered activity sequence of a case, encoded against `activities`.

`activities` character vector of distinct activity labels; the integer code `i` in `sequences` refers to `activities[i]`.

`times` named list of the ordered event timestamps per case.

`case_attributes` data frame of case-level attributes, one row per case, row names equal to the case identifiers.

Examples

```
log <- data.frame(
  case = c("c1", "c1", "c2", "c2"),
  act  = c("register", "decide", "register", "reject"),
  ts   = as.POSIXct("2020-01-01") + c(0, 60, 0, 90),
  status = c("open", "open", "closed", "closed")
)
as_traces(log, case_id = "case", activity = "act", timestamp = "ts")
```

`augment_log` *Join cluster memberships onto an event log*

Description

`augment_log()` adds a `.cluster` column to an event log, mapping every event to the cluster of its case.

Usage

```
augment_log(clust, log)
```

Arguments

`clust`	An `outrank_clust` object from `cluster_traces()`.
`log`	The event-log `data.frame` the clustering came from.

Details

The case-id column is detected as the column of `log` whose values cover all clustered case ids; a message reports which column was used.

Value

log with an added integer .cluster column (NA for cases absent from the clustering, e.g. trimmed outliers).

Examples

```
log <- data.frame(case = rep(c("a", "b", "c", "d"), each = 2),
                  act = rep(c("x", "y"), 4),
                  ts = as.POSIXct("2020-01-01") + (0:7) * 60)
m <- matrix(0.1, 4, 4); m[1:2, 1:2] <- 0.9; m[3:4, 3:4] <- 0.9
diag(m) <- 1; dimnames(m) <- list(c("a", "b", "c", "d"), c("a", "b", "c", "d"))
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.95)
tr <- as_traces(log, "case", "act", "ts")
clust <- cluster_traces(outrank_similarity(tr, crit), k = 2, seed = 1)
augment_log(clust, log)
```

---

cluster_traces *Cluster traces from a credibility matrix*

Description

cluster_traces() partitions the cases of an outrank_similarity() result into k groups, by normalized spectral clustering or hierarchical clustering.

Usage

```
cluster_traces(
  sim,
  k,
  method = c("spectral", "hierarchical"),
  nstart = 100,
  seed = NULL,
  hclust_method = "ward.D2"
)
```

Arguments

sim	An outrank_sim object from outrank_similarity().
k	Number of clusters (an integer, 2 <= k <= n).
method	"spectral" (default) or "hierarchical".
nstart	Number of k-means restarts for the spectral method.
seed	Optional integer seed for the k-means randomness; set it for reproducible spectral memberships. The global RNG state is restored on exit.
hclust_method	Linkage for the hierarchical method; passed to stats::hclust(). Defaults to "ward.D2".

Details

Spectral clustering follows Ng, Jordan and Weiss (2002). With the affinity `S` (diagonal zeroed) and `D = diag(rowSums(S))`, it forms the symmetric normalized Laplacian $L_{sym} = I - D^{-1/2} S D^{-1/2}$, takes the eigenvectors of its `k` smallest eigenvalues, normalises each row to unit length, and runs k-means (with `nstart` restarts) on the result.

Hierarchical clustering runs `stats::hclust()` on the dissimilarity `as.dist(1 - S)` and cuts the tree at `k` groups.

Value

An object of class `outrank_clust`: a list with the integer `memberships` (named by case id), `k`, `method`, the `sim` object, the Laplacian `eigenvalues` (spectral only), and the `hclust` tree (hierarchical only).

References

Ng, A., Jordan, M. and Weiss, Y. (2002). On spectral clustering: analysis and an algorithm. *NIPS*.

Examples

```
m <- matrix(c(1, 0.9, 0.1, 0.1,
              0.9, 1, 0.1, 0.1,
              0.1, 0.1, 1, 0.9,
              0.1, 0.1, 0.9, 1), 4, 4,
            dimnames = list(1:4, 1:4))
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.95)
tr <- as_traces(
  data.frame(case = as.character(1:4), act = "x",
             ts = as.POSIXct("2020-01-01")),
  "case", "act", "ts"
)
sim <- outrank_similarity(tr, crit)
cluster_traces(sim, k = 2, seed = 1)
```

---

constraints *Must-link and cannot-link constraints*

Description

Inject domain knowledge into the credibility matrix by rewarding pairs that should share a cluster (`must_link()`) or severing pairs that should not (`cannot_link()`), following Delias et al. (2023).

Usage

```
must_link(sim, pairs_or_attribute, reward = 2)

cannot_link(sim, pairs_or_attribute)
```

Arguments

`sim`	An `outrank_sim` object from `outrank_similarity()`.
`pairs_or_attribute`	A two-column matrix of case-id pairs, or a length-one attribute name (requires that `sim` carries case attributes).
`reward`	Positive amount added to `S` for each must-linked pair.

Details

With a symmetric 0/1 constraint mask `M` (zero diagonal),

$$\text{must\_link:} \quad S \leftarrow \min(S + \text{reward} \cdot M,\ 1),$$

$$\text{cannot\_link:} \quad S \leftarrow S \odot (1 - M).$$

The constraint set is given either as a two-column matrix of case-id pairs, or as the name of a case attribute: `must_link()` then links cases with an equal attribute value, and `cannot_link()` severs cases whose values differ.

Value

The `outrank_sim` with an updated `S`.

Examples

```
m <- matrix(0.2, 4, 4); diag(m) <- 1; dimnames(m) <- list(1:4, 1:4)
crit <- criterion(m, "similarity", indifference = 0.1, similarity = 0.9)
tr <- as_traces(data.frame(case = as.character(1:4), act = "x",
                           ts = as.POSIXct("2020-01-01")),
                "case", "act", "ts")
sim <- outrank_similarity(tr, crit)
sim <- must_link(sim, cbind("1", "2"))
cannot_link(sim, cbind("3", "4"))$S
```

`crit_attribute` *Attribute-based criterion templates*

Description

Criterion constructors for case-level attributes, covering the common measurement scales. Each builds a measure from a named attribute of the `traces` object.

Usage

```
crit_nominal(attribute, weight = 1, name = attribute)

crit_ordinal(
  attribute,
  levels,
  weight = 1,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
  veto = NULL,
  name = attribute
)

crit_numeric(
  attribute,
  weight = 1,
  transform = identity,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
  veto = NULL,
  name = attribute
)
```

Arguments

`attribute`	Name of a case attribute carried by the `traces` object.
`weight`	A single positive weight.
`name`	Criterion label; defaults to the attribute name.
`levels`	For `crit_ordinal()`, the attribute values from lowest to highest rank.
`indifference`, `similarity`, `veto`	Thresholds; a number or an `as_quantile()` specification. Defaults suit the direction and are overridable.
`transform`	For `crit_numeric()`, a function applied to the numeric attribute before differencing (e.g. `log`); defaults to `identity()`.

Details

- `crit_nominal()` scores a pair 1 when the attribute is equal and 0 otherwise (similarity direction; thresholds preset, no veto).
- `crit_ordinal()` uses the absolute rank difference `|rank_a - rank_b|` over the ordered `levels` (dissimilarity direction); it covers Likert scales.
- `crit_numeric()` uses `|x_a - x_b|` after an optional `transform` (dissimilarity direction); it covers quantitative, interval and percentage scales.

Value

A `criterion()` object.

See Also

crit_activity_profile() and the other process-aware helpers.

Examples

```
crit_nominal("status", weight = 0.3)
crit_ordinal("triage", levels = c("green", "yellow", "red"), weight = 0.2)
crit_numeric("age", weight = 0.2, transform = identity)
```

---

crit_custom *Custom criterion*

Description

The escape hatch for criteria that the templates do not cover, including composite measures (e.g. a linear combination of others). Wraps criterion() directly.

Usage

```
crit_custom(
  x,
  direction,
  weight = 1,
  indifference,
  similarity,
  veto = NULL,
  name = "custom"
)
```

Arguments

x	A function(traces) returning a symmetric numeric matrix, or a precomputed numeric matrix with case-id dimnames.
direction	"similarity" or "dissimilarity".
weight	A single positive weight.
indifference, similarity	Required thresholds (number or as_quantile()).
veto	Optional veto threshold.
name	Criterion label.

Value

A criterion() object.

Examples

```
# A criterion on a precomputed similarity matrix.
m <- matrix(c(1, 0.7, 0.7, 1), 2, dimnames = list(c("a", "b"), c("a", "b")))
crit_custom(m, direction = "similarity", weight = 1,
            indifference = 0.4, similarity = 0.9)
```

`crit_process` *Process-aware criteria*

Description

Convenience constructors that build a `criterion()` from a trace measure, with sensible, overridable quantile-threshold defaults. The similarity measures (`crit_activity_profile()`, `crit_transitions()`) point in the `"similarity"` direction; the distance measures point in the `"dissimilarity"` direction.

Usage

```
crit_activity_profile(
  weight = 1,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.8),
  veto = NULL,
  name = "activity_profile"
)

crit_transitions(
  weight = 1,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.8),
  veto = NULL,
  name = "transitions"
)

crit_edit_distance(
  weight = 1,
  method = "osa",
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
  veto = NULL,
  name = "edit_distance"
)

crit_trace_length(
  weight = 1,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
```

```
  veto = NULL,
  name = "trace_length"
)

crit_distinct_activities(
  weight = 1,
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
  veto = NULL,
  name = "distinct_activities"
)

crit_duration(
  weight = 1,
  units = "mins",
  indifference = as_quantile(0.5),
  similarity = as_quantile(0.2),
  veto = NULL,
  name = "duration"
)
```

Arguments

`weight`	A single positive weight (normalised later, when the similarity matrix is built).
`indifference`, `similarity`, `veto`	Thresholds, each a number or an `as_quantile()` quantile specification. The defaults differ by direction and are documented in the argument defaults.
`name`	Criterion label.
`method`	Edit-distance method passed to `stringdist::seq_distmatrix()`.
`units`	Time unit for durations, passed to `base::difftime()`.

Details

- `crit_activity_profile()` – cosine similarity of activity count vectors.
- `crit_transitions()` – cosine similarity of distance-weighted transition profiles (weight `1 / gap`), as in the foundations paper.
- `crit_edit_distance()` – OSA edit distance on the integer-encoded traces.
- `crit_trace_length()` – `|n_a - n_b|`, the difference in event counts.
- `crit_distinct_activities()` – difference in the number of distinct activities.
- `crit_duration()` – difference in case duration.

Value

A `criterion()` object.

See Also

crit_nominal(), crit_ordinal(), crit_numeric(), crit_custom() for attribute-based criteria.

Examples

```
crit_activity_profile(weight = 0.2)
crit_edit_distance(weight = 0.2, similarity = 2, indifference = 3, veto = 6)
```

---

criterion *Define an outranking criterion*

Description

criterion() is the core constructor behind the crit_* helpers; users rarely call it directly. It bundles a pairwise measure with its direction, weight and ELECTRE-III-style thresholds.

Usage

```
criterion(
  measure,
  direction = c("similarity", "dissimilarity"),
  weight = 1,
  indifference,
  similarity,
  veto = NULL,
  name = NULL
)
```

Arguments

measure	Either a function(traces) returning a symmetric numeric matrix, or a precomputed numeric matrix with dimnames matching the case ids.
direction	One of "similarity" (larger values mean more alike) or "dissimilarity" (smaller values mean more alike).
weight	A single positive number. Weights are normalised to sum to one when the credibility matrix is built.
indifference, similarity	Required thresholds; each a single number or a as_quantile() quantile specification.
veto	Optional veto threshold; a number, an as_quantile() specification, or NULL for no discordance.
name	Optional label used in printing and diagnostics.

Details

For a criterion with indifference threshold q, similarity threshold s and (optional) veto threshold t, and a pairwise value $v = g(a, b)$, the partial concordance and discordance of a `"similarity"`-direction criterion are

$$c = 1 \text{ if } v \geq s, \quad c = \frac{v - q}{s - q} \text{ if } q \leq v < s, \quad c = 0 \text{ if } v < q,$$

$$d = 1 \text{ if } v \leq t, \quad d = \frac{q - v}{q - t} \text{ if } t < v \leq q, \quad d = 0 \text{ if } v > q.$$

For a `"dissimilarity"`-direction criterion the inequalities reverse. This imposes an ordering on the thresholds, which the constructor validates: $t < q < s$ for the similarity direction and $s < q < t$ for the dissimilarity direction. A `veto = NULL` criterion contributes no discordance.

Thresholds may be given as plain numbers or as a quantile specification `as_quantile()`. Numeric thresholds are validated immediately; quantile thresholds are resolved and validated once the measure matrix is known.

Value

An object of class `criterion`.

Examples

```
# A similarity criterion on a precomputed matrix.
m <- matrix(c(1, 0.4, 0.4, 1), 2, dimnames = list(c("a", "b"), c("a", "b")))
criterion(m, direction = "similarity", weight = 2,
          indifference = 0.5, similarity = 0.8, veto = 0.2)
```

---

eigengap *Eigenvalue gap diagnostic*

Description

`eigengap()` plots the smallest eigenvalues of the symmetric normalized Laplacian of `S`. A pronounced gap after the `k`-th eigenvalue suggests `k` well-separated clusters.

Usage

```
eigengap(sim, k_max = 30)
```

Arguments

`sim`	An `outrank_sim` object from `outrank_similarity()`.
`k_max`	Number of smallest eigenvalues to show.

Value

Invisibly, a data frame with the eigenvalue `index`, the `eigenvalue`, and the `gap` to the next one.

Examples

```
m <- matrix(0.1, 6, 6) + diag(0.9, 6)
m[1:3, 1:3] <- 0.9; m[4:6, 4:6] <- 0.9; diag(m) <- 1
dimnames(m) <- list(1:6, 1:6)
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.95)
tr <- as_traces(
  data.frame(case = as.character(1:6), act = "x",
             ts = as.POSIXct("2020-01-01")),
  "case", "act", "ts"
)
eigengap(outrank_similarity(tr, crit), k_max = 5)
```

---

illustrative_log *Illustrative customer-service event log*

Description

The synthetic event log of Table 1 in Delias et al. (2023), used to illustrate outranking-based trace clustering. It describes 25 fictitious customers of a service desk with two tiers, `"GOLD"` and `"NORMAL"` (the latter is the "Blue" tier of the paper's narrative). Cases are numbered in Table 1's order; the last two (case ids `"24"` and `"25"`) are deliberate outliers whose flow matches neither tier.

Usage

```
illustrative_log
```

Format

A data frame with 117 rows (events) and 5 columns:

case_id Customer identifier, `"1"`–`"25"` (character).

activity Activity performed (`A`–`E`), in chronological order.

timestamp Event time (`POSIXct`); events are one minute apart.

status Customer tier, `"GOLD"` or `"NORMAL"`.

satisfaction Registered satisfaction, `"High"` or `"Low"`.

Source

Delias, P., Doumpos, M., Grigoroudis, E. and Matsatsinis, N. (2023). Improving the non-compensatory trace-clustering decision process. *International Transactions in Operational Research*, 30(3), 1387-1406. doi:10.1111/itor.13062 (Table 1).

`outrank_similarity` *Outranking credibility matrix*

Description

`outrank_similarity()` aggregates a set of criteria into the ELECTRE-III credibility matrix `S`, the pairwise similarity used for clustering.

Usage

```
outrank_similarity(traces, criteria, keep_partials = FALSE)
```

Arguments

`traces`	A `traces` object (see `as_traces()`); defines the case set and their order.
`criteria`	A `criterion()` object, or a non-empty list of them.
`keep_partials`	If `TRUE`, retain the per-criterion measure, concordance and discordance matrices for inspection.

Details

Each criterion contributes a partial concordance `c_j` and discordance `d_j` (see `criterion()`). With weights `w_j` normalised to sum to one, the aggregation is

$$C(a,b) = \sum_j w_j \, c_j(a,b),$$

$$D(a,b) = 1 - \prod_{j \in J(a,b)} \frac{1-d_j}{1-c_j}, \quad J(a,b) = \{j : d_j(a,b) > c_j(a,b)\},$$

$$S(a,b) = \min\big(C(a,b),\, 1 - D(a,b)\big).$$

A criterion with `c_j = 1` is never in `J` (concordance is already maximal), which also avoids the division by `1 - c_j`.

Quantile thresholds (`as_quantile()`) are resolved against the off-diagonal distribution of each criterion's own measure matrix before the indices are computed. Weights are normalised to sum to one, with a message when they did not already.

Value

An object of class `outrank_sim`: a list with the credibility matrix `S`, the aggregate concordance `C` and discordance `D`, the case ids, the resolved criteria with normalised weights, and (optionally) the partial matrices.

Examples

```
log <- data.frame(
  case = rep(c("a", "b", "c"), each = 2),
  act  = c("x", "y", "x", "y", "x", "x"),
  ts   = as.POSIXct("2020-01-01") + c(0, 1, 0, 1, 0, 1) * 60
)
tr <- as_traces(log, "case", "act", "ts")
# A criterion on a precomputed similarity matrix (crit_* helpers wrap this).
m <- matrix(c(1, 0.8, 0.2, 0.8, 1, 0.3, 0.2, 0.3, 1), 3, 3,
            dimnames = list(c("a", "b", "c"), c("a", "b", "c")))
crit <- criterion(m, direction = "similarity",
                  indifference = 0.4, similarity = 0.9, veto = 0.1)
outrank_similarity(tr, crit)
```

---

plot.outrank_clust *Plot a trace clustering*

Description

Diagnostic plots for an `outrank_clust` object.

Usage

```
## S3 method for class 'outrank_clust'
plot(
  x,
  type = c("dendrogram", "eigenvalues", "profile"),
  attribute = NULL,
  ...
)
```

Arguments

`x`	An `outrank_clust` object from `cluster_traces()`.
`type`	One of `"dendrogram"` (the hierarchical tree; built on `as.dist(1 - S)` when the clustering was spectral), `"eigenvalues"` (the smallest normalized-Laplacian eigenvalues), or `"profile"` (the distribution of a case `attribute` across clusters).
`attribute`	For `type = "profile"`, the name of a case attribute carried by the underlying `sim`.
`...`	Passed to the underlying plotting call.

Value

`x`, invisibly.

Examples

```
m <- matrix(0.1, 6, 6); m[1:3, 1:3] <- 0.9; m[4:6, 4:6] <- 0.9
diag(m) <- 1; dimnames(m) <- list(1:6, 1:6)
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.95)
tr <- as_traces(data.frame(case = as.character(1:6), act = "x",
                           ts = as.POSIXct("2020-01-01")),
                "case", "act", "ts")
clust <- cluster_traces(outrank_similarity(tr, crit), k = 2, seed = 1)
plot(clust, type = "eigenvalues")
```

---

summary.traces *Per-case summary of a* traces *object*

Description

Per-case summary of a traces object

Usage

```
## S3 method for class 'traces'
summary(object, ...)
```

Arguments

object	A traces object.
...	Unused.

Value

A data frame with one row per case giving the number of events and the number of distinct activities.

---

trim_outliers *Trim outlier cases*

Description

trim_outliers() removes a fraction of the least-connected cases from an outrank_similarity() result, before clustering.

Usage

```
trim_outliers(sim, prop = 0.05, method = c("greedy", "lp"), lp_max = 150L)
```

Arguments

sim	An outrank_sim object from outrank_similarity().
prop	Fraction of cases to remove (in [0, 1)).
method	"greedy" (default) or "lp".
lp_max	Size above which the "lp" method warns about its cost.

Details

Let k = floor(prop * n). The "greedy" method removes the k cases with the smallest total similarity to the others (row sums of S with the diagonal excluded). The "lp" method solves the integer linear program of Delias et al. (2023),

$$\min_{o,r} \sum_{i,j} s_{ij} r_{ij} \quad \text{s.t.} \quad \sum_{i} o_i = k,\ o_i \le r_{ij},\ o_i \le r_{ji},\ o, r \in \{0,1\},$$

which selects the k cases whose incident edges carry the least similarity. A small constant is added to S so that zero entries do not leave r unconstrained. The LP has n + n^2 binary variables and is intended for small n; it warns above lp_max.

Value

A trimmed outrank_sim, with a trimmed attribute giving the removed case ids.

References

Delias, P., Doumpos, M., Grigoroudis, E. and Matsatsinis, N. (2023). Improving the non-compensatory trace-clustering decision process. *International Transactions in Operational Research*, 30(3), 1387-1406. doi:10.1111/itor.13062

Examples

```
m <- matrix(0.8, 5, 5); m[5, ] <- 0.05; m[, 5] <- 0.05; diag(m) <- 1
dimnames(m) <- list(1:5, 1:5)
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.9)
tr <- as_traces(data.frame(case = as.character(1:5), act = "x",
                           ts = as.POSIXct("2020-01-01")),
                "case", "act", "ts")
sim <- outrank_similarity(tr, crit)
trim_outliers(sim, prop = 0.2)
```

---

validate_clusters *Internal cluster validity indices*

Description

validate_clusters() reports the connectivity and Dunn index of a clustering, computed on the dissimilarity as.dist(1 - S).

Usage

```
validate_clusters(clust)
```

Arguments

clust	An outrank_clust object from cluster_traces().

Details

Connectivity (to be minimised) and the Dunn index (to be maximised) are obtained from the **clValid** package. If **clValid** is not installed, the function returns NA values with an informative message.

Value

A named numeric vector with connectivity and dunn.

Examples

```
m <- matrix(0.1, 6, 6); m[1:3, 1:3] <- 0.9; m[4:6, 4:6] <- 0.9
diag(m) <- 1; dimnames(m) <- list(1:6, 1:6)
crit <- criterion(m, "similarity", indifference = 0.3, similarity = 0.95)
tr <- as_traces(data.frame(case = as.character(1:6), act = "x",
                           ts = as.POSIXct("2020-01-01")),
                "case", "act", "ts")
clust <- cluster_traces(outrank_similarity(tr, crit), k = 2, seed = 1)
validate_clusters(clust)
```

Index