Least Squares Fit of Additive Tree Distances to Dissimilarities

Description

Find the additive tree distance or centroid distance minimizing least squares distance (Euclidean dissimilarity) to a given dissimilarity object.

Usage

```
ls_fit_addtree(x, method = c("SUMT", "IP", "IR"), weights = 1,
               control = list())
ls_fit_centroid(x)
```

Arguments

`x`	a dissimilarity object inheriting from class `"dist"`.
`method`	a character string indicating the fitting method to be employed. Must be one of `"SUMT"`(default), `"IP"`, or `"IR"`, or a unique abbreviation thereof.
`weights`	a numeric vector or matrix with non-negative weights for obtaining a weighted least squares fit. If a matrix, its numbers of rows and columns must be the same as the number of objects in `x`, and the lower diagonal part is used. Otherwise, it is recycled to the number of elements in `x`.
`control`	a list of control parameters. See **Details**.

Details

See `as.cl_addtree` for details on additive tree distances and centroid distances.

With *L(d) = ∑ w_{ij} (x_{ij} - d_{ij})^2*, the problem to be solved by `ls_fit_addtree` is minimizing *L*over all additive tree distances *d*. This problem is known to be NP hard.

We provide three heuristics for solving this problem.

Method `"SUMT"` implements the SUMT (Sequential Unconstrained Minimization Technique, Fiacco and McCormick, 1968) approach of de Soete (1983). Incomplete dissimilarities are currently not supported.

Methods `"IP"` and `"IR"` implement the Iterative Projection and Iterative Reduction approaches of Hubert and Arabie (1995) and Roux (1988), respectively. Non-identical weights and incomplete dissimilarities are currently not supported.

See `ls_fit_ultrametric` for details on these methods and available control parameters.

It should be noted that all methods are heuristics which cannot be guaranteed to find the global minimum. Standard practice would recommend to use the best solution found in “sufficiently many” replications of the base algorithm.

`ls_fit_centroid` finds the centroid distance *d* minimizing *L(d)* (currently, only for the case of identical weights). This optimization problem has a closed-form solution.

Value

An object of class `"cl_addtree"` containing the optimal additive tree distances.

References

A. V. Fiacco and G. P. McCormick (1968). *Nonlinear programming: Sequential unconstrained minimization techniques*. New York: John Wiley & Sons.

L. Hubert and P. Arabie (1995). Iterative projection strategies for the least squares fitting of tree structures to proximity data. *British Journal of Mathematical and Statistical Psychology*, **48**, 281–317. doi: 10.1111/j.2044-8317.1995.tb01065.x.

M. Roux (1988). Techniques of approximation for building two tree structures. In C. Hayashi and E. Diday and M. Jambu and N. Ohsumi (Eds.), *Recent Developments in Clustering and Data Analysis*, pages 151–170. New York: Academic Press.

G. de Soete (1983). A least squares algorithm for fitting additive trees to proximity data. *Psychometrika*,**48**, 621–626. doi: 10.1007/BF02293884.