Package 'aramappings'

November 26, 2025

Title Computes Adaptable Radial Axes Mappings

Version 0.1.2

Description Computes low-dimensional point representations of high-dimensional numerical data according to the data visualization method Adaptable Radial Axes described in: Manuel Rubio-Sánchez, Alberto Sanchez, and Dirk J. Lehmann (2017) ``Adaptable radial axes plots for improved multivariate data visualization" <doi:10.1111/cgf.13196>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports clarabel, CVXR, glpkAPI, Matrix, parallel, pracma, Rglpk, slam, ggplot2, grDevices, grid, stats

URL https://github.com/manuelrubio/aramappings, https://manuelrubio.github.io/aramappings/

BugReports https://github.com/manuelrubio/aramappings/issues

Suggests testthat (>= 3.0.0), parallelly, geometry, ascentTraining, liver, datasetsICR, knitr, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Manuel Rubio-Sánchez [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-8692-2553>), Dirk J. Lehmann [ctb] (ORCID: <https://orcid.org/0000-0002-8089-4408>), Miguel Ángel Muñoz Mohedano [ctb] (ORCID: <https://orcid.org/0000-0001-6114-4460>), Alberto Sánchez Campos [ctb] (ORCID: <https://orcid.org/0000-0002-5382-6805>), Cristina Soguero-Ruiz [ctb] (ORCID: <https://orcid.org/0000-0001-5817-989X>)

Maintainer Manuel Rubio-Sánchez <manuel.rubio@urjc.es>

Repository CRAN

Date/Publication 2025-11-26 03:40:07 UTC

Contents

`ara_exact_l1` *Exact Adaptable Radial Axes (ARA) mappings using the L1 norm*

Description

`ara_exact_l1()` computes **exact Adaptable Radial Axes** (ARA) mappings for the **L1 norm**

Usage

```
ara_exact_l1(
  X,
  V,
  variable = 1,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = NULL
)
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.
`solver`	String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".
`use_glpkAPI_simplex`	Boolean parameter that indicates whether to use the simplex algorithm (if `TRUE`) or an interior point method (if `FALSE`), when using the **glpkAPI** solver. The default is `TRUE`.

`cluster` Optional cluster object related to the parallel package. If supplied, and `n_LP_problems` is N, the method computes the mappings using parallel processing.

Details

`ara_exact_l1()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (13), for the L1 vector norm. Its equality constraint forces estimates to be exact for one of the data variables.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

If the chosen solver fails to map one or more data observations (i.e., fails to solve the related optimization problems), their rows in `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
```

```
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Detect the number of available CPU cores
NCORES <- parallelly::availableCores(omit = 1)

# Create a cluster for parallel processing
cl <- parallel::makeCluster(NCORES)

# Compute the mapping
mapping <- ara_exact_l1(
  Z,
  V,
  variable = variable,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = cl
)

# Stop cluster
parallel::stopCluster(cl)

# Select variables with labeled axis lines on ARA plot
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_exact_l2` *Exact Adaptable Radial Axes (ARA) mappings using the L2 norm*

Description

`ara_exact_l2()` computes **exact Adaptable Radial Axes** (ARA) mappings for the **L2 norm**

Usage

```
ara_exact_l2(X, V, variable = 1, solver = "formula")
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.
`solver`	String indicating a package or method for solving the optimization problem. It can be "formula" (default), where the solution is obtained through a closed-form formula, or "CVXR".

Details

`ara_exact_l2()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (13), for the squared-Euclidean norm. Its equality constraint forces estimates to be exact for one of the data variables. The problem admits closed-form solutions.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

The output status vector returns the 2-norm condition number of `V`. If the chosen solver fails to map the data (i.e., fails to solve the related optimization problem), `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Compute the mapping
mapping <- ara_exact_l2(
```

```
  Z,
  V,
  variable = variable,
  solver = "formula"
)

# Select variables with labeled axis lines on ARA plot
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_exact_linf` *Exact Adaptable Radial Axes (ARA) mappings using the L-infinity norm*

Description

`ara_exact_linf()` computes **exact Adaptable Radial Axes** (ARA) mappings for the **L-infinity norm**

Usage

```
ara_exact_linf(
  X,
  V,
  variable = 1,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = NULL
)
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.

`solver` String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".

`use_glpkAPI_simplex` Boolean parameter that indicates whether to use the simplex algorithm (if `TRUE`) or an interior point method (if `FALSE`), when using the **glpkAPI** solver. The default is `TRUE`.

`cluster` Optional cluster object related to the parallel package. If supplied, and `n_LP_problems` is N, the method computes the mappings using parallel processing.

Details

`ara_exact_linf()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (13), for the L-infinity vector norm. Its equality constraint forces estimates to be exact for one of the data variables.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

If the chosen solver fails to map one or more data observations (i.e., fails to solve the related optimization problems), their rows in `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
```

```
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Detect the number of available CPU cores
NCORES <- parallelly::availableCores(omit = 1)

# Create a cluster for parallel processing
cl <- parallel::makeCluster(NCORES)

# Compute the mapping
mapping <- ara_exact_linf(
  Z,
  V,
  variable = variable,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = cl
)

# Stop cluster
parallel::stopCluster(cl)

# Select variables with labeled axis lines on ARA plot
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
```

```
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_ordered_l1` *Ordered Adaptable Radial Axes (ARA) mappings using the L1 norm*

Description

`ara_ordered_l1()` computes **ordered Adaptable Radial Axes** (ARA) mappings for the **L1 norm**

Usage

```
ara_ordered_l1(X, V, variable = 1, solver = "clarabel")
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.
`solver`	String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".

Details

`ara_ordered_l1()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (14), for the L1 norm. The inequality constraint ensures that the estimates for a selected variable are ordered in accordance with its original values. In other words, ignoring any ties, the estimate for the data observation with the i-th smallest value will correspond to the i-th smallest estimate.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

If the chosen solver fails to map the data (i.e., fails to solve the related optimization problem), `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
```

```
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Compute the mapping
mapping <- ara_ordered_l1(
  Z,
  V,
  variable = variable,
  solver = "clarabel"
)

# Select variables with labeled axis lines on ARA plot
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_ordered_l2` *Ordered Adaptable Radial Axes (ARA) mappings using the L2 norm*

Description

`ara_ordered_l2()` computes **ordered Adaptable Radial Axes** (ARA) mappings for the **L2 norm**

Usage

```
ara_ordered_l2(X, V, variable = 1, solver = "clarabel")
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.

`solver` String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".

Details

`ara_ordered_l2()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (14), for the squared-Euclidean norm. The inequality constraint ensures that the estimates for a selected variable are ordered in accordance with its original values. In other words, ignoring any ties, the estimate for the data observation with the i-th smallest value will correspond to the i-th smallest estimate.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

If the chosen solver fails to map the data (i.e., fails to solve the related optimization problem), `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
```

```
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Compute the mapping
mapping <- ara_ordered_l2(
  Z,
  V,
  variable = variable,
  solver = "clarabel"
)

# Select variables with labeled axis lines on ARA plot
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_ordered_linf` *Ordered Adaptable Radial Axes (ARA) mappings using the L-infinity norm*

Description

`ara_ordered_linf()` computes **ordered Adaptable Radial Axes** (ARA) mappings for the **Linf norm**

Usage

```
ara_ordered_linf(X, V, variable = 1, solver = "clarabel")
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`variable`	Integer that indicates the variable (in [1,n]) for which the estimates of high-dimensional data will be exact. Default: variable = 1.
`solver`	String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".

Details

`ara_ordered_linf()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the constrained optimization problem in Eq. (14), for the L-infinity norm. The inequality constraint ensures that the estimates for a selected variable are ordered in accordance with its original values. In other words, ignoring any ties, the estimate for the data observation with the i-th smallest value will correspond to the i-th smallest estimate.

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms.

If the chosen solver fails to map the data (i.e., fails to solve the related optimization problem), `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Select variable for exact estimates, and use it for coloring the embedded
# points
n <- nrow(V)
variable <- sample(1:n, 1)

# Compute the mapping
mapping <- ara_ordered_linf(
  Z,
  V,
  variable = variable,
  solver = "clarabel"
)

# Select variables with labeled axis lines on ARA plot
```

```
axis_lines <- variable

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  axis_lines = axis_lines,
  color_variable = variable
)
```

`ara_unconstrained_l1` *Unconstrained Adaptable Radial Axes (ARA) mappings using the L1 norm*

Description

`ara_unconstrained_l1()` computes **unconstrained Adaptable Radial Axes** (ARA) mappings for the **L1 norm**

Usage

```
ara_unconstrained_l1(
  X,
  V,
  weights = rep(1, ncol(X)),
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = NULL
)
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`weights`	Numeric array specifying optional non-negative weights associated with each variable. The function only considers them if they do not share the same value. Default: array of n ones.
`solver`	String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".

`use_glpkAPI_simplex`
: Boolean parameter that indicates whether to use the simplex algorithm (if `TRUE`) or an interior point method (if `FALSE`), when using the **glpkAPI** solver. The default is `TRUE`.

`cluster`
: Optional cluster object related to the parallel package. If supplied, and `n_LP_problems` is N, the method computes the mappings using parallel processing.

Details

`ara_unconstrained_l1()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the unconstrained optimization problem in Eq. (10), for the L1 vector norm. Specifically, it solves equivalent linear problems as described in (11). Optional non-negative weights (`weights`) associated with each data variable can be supplied to solve the problem in Eq. (15).

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms, and ignoring weights.

If the chosen solver fails to map one or more data observations (i.e., fails to solve the related optimization problems), their rows in `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
```

```
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Define weights
weights <- c(1, 0.75, 0.75, 1)

# Detect the number of available CPU cores
NCORES <- parallelly::availableCores(omit = 1)

# Create a cluster for parallel processing
cl <- parallel::makeCluster(NCORES)

# Compute the mapping
mapping <- ara_unconstrained_l1(
  Z,
  V,
  weights = weights,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = cl
)

# Stop cluster
parallel::stopCluster(cl)

# Select variables with labeled axis lines on ARA plot
axis_lines <- c(1, 4) # 1:"mpg", 4:"acceleration")

# Select variable used for coloring embedded points
color_variable <- 1 # "mpg"

# Draw the ARA plot
draw_ara_plot_2d_standardized(
```

```
  Z,
  X,
  V,
  mapping$P,
  weights = weights,
  axis_lines = axis_lines,
  color_variable = color_variable
)
```

`ara_unconstrained_l2` *Unconstrained Adaptable Radial Axes (ARA) mappings using the L2 norm*

Description

`ara_unconstrained_l2()` computes **unconstrained Adaptable Radial Axes** (ARA) mappings for the **L2 norm**

Usage

```
ara_unconstrained_l2(X, V, weights = rep(1, ncol(X)), solver = "formula")
```

Arguments

`X`	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
`V`	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of `V` defines an axis vector.
`weights`	Numeric array specifying optional non-negative weights associated with each variable. The function only considers them if they do not share the same value. Default: array of n ones.
`solver`	String indicating a package or method for solving the optimization problem. It can be "formula" (default), where the solution is obtained through a closed-form formula, or "CVXR".

Details

`ara_unconstrained_l2()` computes low-dimensional point representations of high-dimensional numerical data (`X`) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in `X` through dot products of the mapped points with the axis vectors (rows of `V`). This particular function solves the unconstrained optimization problem in Eq. (10), for the squared-Euclidean norm. Optional non-negative weights (`weights`) associated with each data variable can be supplied to solve the problem in Eq. (15).

Value

A list with the three following entries:

`P` A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

`status` A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

`objval` The numeric objective value associated with the solution to the optimization problem, considering matrix norms, and ignoring weights.

When solver is "formula" this function always produces valid solutions (`P`), since the pseudo-inverse matrix always exists. Thus, the output status vector is not relevant, but is returned in consonance with other adaptable radial axes functions in the package. If **CVRX** were used and failed to map the data observations (i.e., failed to solve the related optimization problem), `P` would be a matrix containing `NA` (not available) values, and `objval` would be also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)
```

```
# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Define weights
weights <- c(1, 0.75, 0.75, 1)

# Compute the mapping
mapping <- ara_unconstrained_l2(
  Z,
  V,
  weights = weights,
  solver = "formula"
)

# Select variables with labeled axis lines on ARA plot
axis_lines <- c(1, 4) # 1:"mpg", 4:"acceleration")

# Select variable used for coloring embedded points
color_variable <- 1 # "mpg"

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  weights = weights,
  axis_lines = axis_lines,
  color_variable = color_variable
)
```

`ara_unconstrained_linf` *Unconstrained Adaptable Radial Axes (ARA) mappings using the L-infinity norm*

Description

`ara_unconstrained_linf()` computes **unconstrained Adaptable Radial Axes** (ARA) mappings for the **L-infinity norm**

Usage

```
ara_unconstrained_linf(
  X,
  V,
  weights = rep(1, ncol(X)),
```

```
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = NULL
)
```

Arguments

X	Numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
V	Numeric matrix defining the axes or "axis vectors". Its dimensions are n x m, where 1<=m<=3 is the dimension of the visualization space. Each row of V defines an axis vector.
weights	Numeric array specifying optional non-negative weights associated with each variable. The function only considers them if they do not share the same value. Default: array of n ones.
solver	String indicating a package for solving the linear problem(s). It can be "clarabel" (default), "glpkAPI", "Rglpk", or "CVXR".
use_glpkAPI_simplex	Boolean parameter that indicates whether to use the simplex algorithm (if TRUE) or an interior point method (if FALSE), when using the **glpkAPI** solver. The default is TRUE.
cluster	Optional cluster object related to the parallel package. If supplied, and n_LP_problems is N, the method computes the mappings using parallel processing.

Details

ara_unconstrained_linf() computes low-dimensional point representations of high-dimensional numerical data (X) according to the data visualization method "Adaptable Radial Axes" (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196), which describes a collection of convex norm optimization problems aimed at minimizing estimates of original values in X through dot products of the mapped points with the axis vectors (rows of V). This particular function solves the unconstrained optimization problem in Eq. (10), for the L-infinity vector norm. Specifically, it solves equivalent linear problems as described in (12). Optional non-negative weights (weights) associated with each data variable can be supplied to solve the problem in Eq. (15).

Value

A list with the three following entries:

P A numeric N x m matrix containing the mapped points. Each row is the low-dimensional representation of a data observation in X.

status A vector of length N where the i-th element contains the status of the chosen solver when calculating the mapping of the i-th data observation. The type of the elements depends on the particular chosen solver.

objval The numeric objective value associated with the solution to the optimization problem, considering matrix norms, and ignoring weights.

If the chosen solver fails to map one or more data observations (i.e., fails to solve the related optimization problems), their rows in `P` will contain `NA` (not available) values. In that case, `objval` will also be `NA`.

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Define weights
weights <- c(1, 0.75, 0.75, 1)

# Detect the number of available CPU cores
NCORES <- parallelly::availableCores(omit = 1)

# Create a cluster for parallel processing
cl <- parallel::makeCluster(NCORES)
```

```
# Compute the mapping
mapping <- ara_unconstrained_linf(
  Z,
  V,
  weights = weights,
  solver = "glpkAPI",
  use_glpkAPI_simplex = TRUE,
  cluster = cl
)

# Stop cluster
parallel::stopCluster(cl)

# Select variables with labeled axis lines on ARA plot
axis_lines <- c(1, 4) # 1:"mpg", 4:"acceleration")

# Select variable used for coloring embedded points
color_variable <- 1 # "mpg"

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  weights = weights,
  axis_lines = axis_lines,
  color_variable = color_variable
)
```

`draw_ara_plot_2d_standardized` *Draws a 2D Adaptable Radial Axes (ARA) plot for standardized data*

Description

Creates a plot associated with an Adaptable Radial Axes (ARA) mapping

Usage

```
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  P,
  weights = rep(1, ncol(Z)),
  axis_lines = NULL,
  color_variable = NULL
)
```

Arguments

Z	Standardized numeric data matrix of dimensions N x n, where N is the number of observations, and n is the number of variables.
X	Original numeric data matrix (before standardizing) of dimensions N x n
V	Numeric matrix of "axis vectors" of dimensions n x 2, where each row of V defines an axis vector.
P	Numeric data matrix of dimensions N x 2 containing the N 2-dimensional representations of the data observations (i.e., the embedded points).
weights	Numeric array specifying non-negative weights associated with each variable. Can also be a 1D matrix. Default: array of n ones.
axis_lines	Array of integer variable indices (in [1,n]) indicating which calibrated axis lines are to be displayed. Default: NULL.
color_variable	Integer (in [1,n]) that indicates the variable used to color the embedded points. Default: NULL.

Details

The function `draw_ara_plot_2d_standardized()` generates a basic two-dimensional plot related to an "Adaptable Radial Axes" (ARA) mapping (M. Rubio-Sánchez, A. Sanchez, and D. J. Lehmann (2017), doi: 10.1111/cgf.13196) for high-dimensional numerical data (`X`) that has been previously standardized (`Z`). The plot displays a set of 2D points (`P`), each representing an observation from the high-dimensional dataset. It also includes a collection of axis vectors (`V`), each corresponding to a specific data variable. If the ARA mapping incorporates weights (`weights`), these axis vectors are colored accordingly to reflect the weighting. For a user-specified subset of variables (`axis_lines`), the function additionally draws axis lines with tick marks that represent values of the selected variables. Users can estimate the values of the high-dimensional data by visually projecting the plotted points orthogonally onto these axes. The plotted points can also be colored according to the values of the variable `color_variable`.

Value

Returns 0 if the function terminates without errors

References

M. Rubio-Sánchez, A. Sanchez, D. J. Lehmann: Adaptable radial axes plots for improved multivariate data visualization. Computer Graphics Forum 36, 3 (2017), 389–399. doi:10.1111/cgf.13196

Examples

```
# Load data
data("auto_mpg", package = "ascentTraining")

# Define subset of (numerical) variables
# 1:"mpg", 4:"horsepower", 5:"weight", 6:"acceleration"
selected_variables <- c(1, 4, 5, 6)

# Retain only selected variables and rename dataset as X
```

```
X <- auto_mpg[, selected_variables] # Select a subset of variables
rm(auto_mpg)

# Remove rows with missing values from X
N <- nrow(X)
rows_to_delete <- NULL
for (i in 1:N) {
  if (sum(is.na(X[i, ])) > 0) {
    rows_to_delete <- c(rows_to_delete, -i)
  }
}
X <- X[rows_to_delete, ]

# Convert X to matrix
X <- apply(as.matrix.noquote(X), 2, as.numeric)

# Standardize data
Z <- scale(X)

# Define axis vectors (2-dimensional in this example)
r <- c(0.8, 1, 1.2, 1)
theta <- c(225, 100, 315, 80) * 2 * pi / 360
V <- geometry::pol2cart(theta, r)

# Define weights
weights <- c(1, 0.75, 0.75, 1)

# Compute the mapping
mapping <- ara_unconstrained_l2(Z, V, weights = weights, solver = "formula")

# Select variables with labeled axis lines on ARA plot
axis_lines <- c(1, 4) # 1:"mpg", 4:"acceleration")

# Select variable used for coloring embedded points
color_variable <- 1 # "mpg"

# Draw the ARA plot
draw_ara_plot_2d_standardized(
  Z,
  X,
  V,
  mapping$P,
  weights = weights,
  axis_lines = axis_lines,
  color_variable = color_variable
)
```

Index