Package 'staRsom'

November 28, 2026

Type Package

Title Self-Organising Maps for EEMs from DOM

Version 0.3.0

Description Self-organising maps (SOM, Kohonen 2013) <doi:10.1016/j.neunet.2012.09.018> for excitation emission matrix (EEM) data of dissolved organic matter (DOM) prepared with 'staRdom'. Samples are arranged on a map, grouped by clustering the prototype EEMs and fluorescence regions are derived from correlated component planes following Ejarque-Gonzalez and Butturini (2014) <doi:10.1371/journal.pone.0099618>. Results can be split into sample groups for subsequent PARAFAC analyses.

License AGPL-3

URL https://matthiaspucher.github.io/staRsom/, https://github.com/MatthiasPucher/staRsom

BugReports https://github.com/MatthiasPucher/staRsom/issues

Encoding UTF-8

Language en-GB

Depends R (>= 4.4.0), staRdom (>= 1.1.32)

Imports cluster, dplyr, eemR, ggplot2, kohonen (>= 3.0.12), parallel, stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Matthias Pucher [aut, cre] (ORCID: <https://orcid.org/0000-0002-9093-2199>)

Maintainer Matthias Pucher <matthias.pucher@ecosight.at>

Repository CRAN

Date/Publication 2026-11-28 14:50:02 UTC

Contents

eem2mat . . . 2
eempf_som . . . 4
eemsom . . . 5
eemsom_cluster . . . 8
eemsom_codes . . . 9
eemsom_fri . . . 10
eemsom_kselect . . . 11
eemsom_plot_cplane . . . 12
eemsom_plot_map . . . 14
eemsom_plot_silhouette . . . 15
eemsom_plot_stability . . . 16
eemsom_plot_training . . . 17
eemsom_predict . . . 18
eemsom_quality . . . 19
eemsom_regions . . . 20
eemsom_samples . . . 22
eemsom_split . . . 23
eemsom_stability . . . 24
ggeem.eemsom . . . 26
ggeem.eemsom_regions . . . 28
mat2eem . . . 29

Index **31**

`eem2mat` *Unfold EEMs into a matrix of samples and wavelength pairs*

Description

Each EEM in an eemlist is unfolded into a vector. The result is a matrix with one row per sample and one column per pair of excitation and emission wavelengths. This matrix is the input for the self-organising map in `eemsom`. The unfolding can be reversed by `mat2eem`.

Usage

```
eem2mat(eem_list, normalise = "sd", na = "remove")
```

Arguments

`eem_list`	object of class eemlist. All EEMs need to have the same excitation and emission wavelengths and unique sample names.
`normalise`	normalisation of each sample: `"sd"` (default) divides each EEM by its standard deviation like `norm_array`, `"max"` by its maximum (Ejarque-Gonzalez and Butturini 2014), `"sum"` by the sum of its fluorescence and `"none"` keeps the original values.

`na` handling of missing values (e.g. removed scatter): `"remove"` (default) removes wavelength pairs that are missing in at least one sample, `"keep"` keeps wavelength pairs that are missing in some of the samples. Wavelength pairs that are missing in all samples are always removed, see details.

Details

Columns are named "ex|em", e.g. "250|450". The EEM matrix is unfolded column-wise, so emission wavelengths change faster than excitation wavelengths.

Normalisation removes differences in concentration and focuses on the shape of the EEMs. The normalisation factors are calculated from the remaining wavelength pairs (after removing missing values) and are saved as an attribute. They need to be positive, so EEMs with a maximum or a sum of zero or below cannot be normalised with `"max"` or `"sum"`.

Wavelength pairs that are missing in all samples (e.g. scatter that was removed but not interpolated) carry no information and are removed even with `na = "keep"`. Otherwise, the self-organising map would contain arbitrary values for them, because kohonen fills missing values in the initial codebook vectors with random values. With `na = "keep"`, kohonen calculates the distances between samples and codebook vectors from the available wavelength pairs and scales them up to the number of all wavelength pairs.

Value

A matrix with samples in rows and wavelength pairs in columns. Attributes: `ex` and `em` (wavelengths of the original EEMs), `keep` (logical vector stating which of the unfolded wavelength pairs are columns of the matrix), `samples`, `normalise` and `norm_factors`.

References

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

`mat2eem`, `eem2array`

Other self-organising maps: `eempf_som()`, `eemsom()`, `mat2eem()`

Examples

```
data(eem_list)

m <- eem2mat(eem_list)
dim(m)
attr(m, "norm_factors")
```

`eempf_som` *Calculate a self-organising map from the loadings of a PARAFAC model*

Description

Instead of the EEMs, the sample loadings (A-mode) of a PARAFAC model are arranged on a self-organising map. The loadings are scaled to Fmax, the maximum fluorescence of each component in each sample. Cuss et al. (2014) found a clearer separation of DOM sources with the relative contributions of PARAFAC components than with the EEMs themselves.

Usage

```
eempf_som(pfmodel, dim = "auto", normalise = TRUE, seed = NULL, ...)

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

Arguments

`pfmodel`	object of class parafac, see `eem_parafac`
`dim`	size of the map, either `"auto"` (default) or a numeric vector `c(xdim, ydim)`, see `eemsom`
`normalise`	logical, if TRUE (default), the Fmax values of each sample are divided by their sum, so the map shows the relative composition of the samples (Cuss et al. 2014, Wheeler et al. 2017). If FALSE, the Fmax values are used directly (Zhang et al. 2015).
`seed`	seed for the random number generator. With `NULL` (default), the current random number stream of the session is used, so `set.seed` before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards.
`...`	additional parameters passed on to `eemsom` (`topo`, `rlen`, `alpha`, `mode`, `init`, `cores`, `verbose`) and to `som`
`x`	object of class eempf_som

Details

The result can be used with the functions for self-organising maps of EEMs, e.g. `eemsom_samples`, `eemsom_cluster`, `eemsom_split` and the plot functions. The prototype EEM of a unit (`eemsom_codes`, `ggeem`) is the sum of the PARAFAC components weighted by the prototype loadings. `eemsom_plot_cplane` shows component planes of the PARAFAC components with the argument `comps`. `eemsom_predict` maps new samples after calculating their loadings with the fixed emission and excitation loadings of the PARAFAC model.

Relative contributions (`normalise = TRUE`) are only meaningful for non-negative loadings, i.e. for models calculated with `const = c("nonneg", "nonneg", "nonneg")` in `eem_parafac`.

`eemsom_regions` and `eemsom_stability` are not available for these maps.

Value

object of class eempf_som and eemsom, see eemsom. Additionally, it contains the element pf-model, the PARAFAC model with emission (B) and excitation (C) loadings scaled to Fmax (see eempf_rescaleBC), which is used to reconstruct prototype EEMs and to map new samples. The element normalise is the logical value of the argument and norm_factors are the sums of the Fmax values (or 1).

References

Cuss, C. W., Shi, Y. X., McConnell, S. M., Gueguen, C. (2014). Changes in the fluorescence composition of multiple DOM sources over pH gradients assessed by combining parallel factor analysis and self-organizing maps. Journal of Geophysical Research: Biogeosciences 119(9), 1850-1860. doi:10.1002/2013JG002598

Wheeler, K. I., Levia, D. F., Hudson, J. E. (2017). Tracking senescence-induced patterns in leaf litter leachate using parallel factor analysis (PARAFAC) modeling and self-organizing maps. Journal of Geophysical Research: Biogeosciences 122(9), 2233-2250. doi:10.1002/2016JG003677

Zhang, Y., Liang, X., Wang, Z., Xu, L. (2015). A novel approach combining self-organizing map and parallel factor analysis for monitoring water quality of watersheds under non-point source pollution. Scientific Reports 5, 16079. doi:10.1038/srep16079

See Also

eemsom, eem_parafac

Other self-organising maps: eem2mat(), eemsom(), mat2eem()

Examples

```
data(pf_models)

pfsom <- eempf_som(pf4[[1]], rlen = 100, seed = 42)
pfsom
eemsom_plot_cplane(pfsom, comps = 1:6)
pfsom <- eemsom_cluster(pfsom, k = 4)
eemsom_plot_map(pfsom, type = "cluster")
ggeem(pfsom, units = 1:4)
```

---

eemsom *Calculate a self-organising map from EEM data*

Description

The EEMs are unfolded into vectors (see eem2mat), normalised and arranged on a self-organising map (SOM, Kohonen 2013). Each unit of the map gets a prototype EEM (codebook vector). Similar samples are mapped to the same or neighbouring units. The calculation is done with som, please see details there.

Usage

```
eemsom(
  eem_list,
  dim = "auto",
  topo = "hexagonal",
  normalise = "sd",
  na = "remove",
  rlen = 500,
  alpha = c(0.05, 0.01),
  mode = "online",
  init = "pca",
  seed = NULL,
  cores = 1,
  verbose = FALSE,
  ...
)

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

Arguments

eem_list	object of class eemlist. All EEMs need to have the same excitation and emission wavelengths and unique sample names.
dim	size of the map, either "auto" (default) or a numeric vector c(xdim, ydim) with the number of columns and rows of units
topo	topology of the map, "hexagonal" (default) or "rectangular"
normalise	normalisation of each sample, see eem2mat
na	handling of missing values, see eem2mat
rlen	number of times the complete data set is presented to the map during training
alpha	learning rate, vector of start and end value
mode	training algorithm: "online" (default), "batch" or "pbatch" (parallel batch)
init	initialisation of the codebook vectors: "pca" (default) places them on the plane of the first two principal components, "random" uses randomly selected samples
seed	seed for the random number generator. With NULL (default), the current random number stream of the session is used, so set.seed before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards.
cores	number of cores used with mode = "pbatch"
verbose	print infos
...	additional parameters that are passed on to som, e.g. radius. keep.data is always TRUE, because the data are needed by eemsom_samples and other functions.
x	object of class eemsom

Details

With `dim = "auto"`, the number of units is $5\sqrt{n}$ (Vesanto and Alhoniemi 2000) but not more than the number of samples n. The ratio of the side lengths follows the two largest eigenvalues of the data (Bieroza et al. 2009, Ejarque-Gonzalez and Butturini 2014): like in the SOM Toolbox (Vesanto et al. 2000), it is the square root of the ratio of the eigenvalues, i.e. the ratio of the standard deviations along the first two principal components. Ratios above the square root of the number of units are replaced by 1 to avoid very elongated maps.

The default learning rate (0.05 to 0.01) and the default neighbourhood radius (decreasing from two thirds of all unit distances to zero) of `som` correspond to the settings of Ejarque-Gonzalez and Butturini (2014).

The squared Euclidean distance (`dist.fcts = "sumofsquares"`) is always used. Otherwise, kohonen would switch to the Tanimoto distance for data between 0 and 1 with row sums of 1, which occurs with `normalise = "sum"`. With `na = "keep"`, distances are calculated from the available wavelength pairs of a sample and scaled up to the number of all wavelength pairs.

The initialisation with `init = "pca"` is deterministic. The order of samples in the online training is random, so please set `seed` or use `set.seed` before the call for reproducible results.

The training time grows with the numbers of samples, units and wavelength pairs. As a rough guide for one core, online training with `rlen = 500` takes about a minute for 200 EEMs with 3000 wavelength pairs and several minutes for 1000 EEMs. `mode = "pbatch"` with `cores > 1`, a smaller `rlen` or a lower resolution of the EEMs (see `eem_range` and `eem_exclude`; Ejarque-Gonzalez and Butturini (2014) used 366 wavelength pairs) reduce the time. This matters for `eemsom_stability`, which repeats the training many times.

Value

object of class eemsom, a list containing

- som: object of class kohonen, see `som`
- ex, em: excitation and emission wavelengths
- keep: logical vector of the unfolded wavelength pairs used in the SOM
- samples: sample names
- normalise, norm_factors: normalisation method and factors
- na: handling of missing values
- clusters: cluster of each unit, NULL until the units are clustered
- settings: arguments used for `som`, needed by `eemsom_stability`

References

Kohonen, T. (2013). Essentials of the self-organizing map. Neural Networks 37, 52-65. doi:10.1016/j.neunet.2012.09.018

Vesanto, J., Alhoniemi, E. (2000). Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586-600. doi:10.1109/72.846731

Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J. (2000). SOM Toolbox for Matlab 5. Report A57, Helsinki University of Technology.

Bieroza, M., Baker, A., Bridgeman, J. (2009). Exploratory analysis of excitation-emission matrix fluorescence spectra with self-organizing maps as a basis for determination of organic matter removal efficiency at water treatment works. Journal of Geophysical Research: Biogeosciences 114(G4), 2009JG000940. doi:10.1029/2009JG000940

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

Wehrens, R., Kruisselbrink, J. (2018). Flexible Self-Organizing Maps in kohonen 3.0. Journal of Statistical Software 87(7), 1-18. doi:10.18637/jss.v087.i07

See Also

som, eemsom_samples, eemsom_codes, eemsom_quality

Other self-organising maps: eem2mat(), eempf_som(), mat2eem()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
som
eemsom_samples(som)
```

---

eemsom_cluster *Cluster the units of a self-organising map*

Description

The prototypes (codebook vectors) of the units are grouped by hierarchical clustering or k-means. Samples belong to the cluster of their unit. This two-level approach (Vesanto and Alhoniemi 2000) is faster and more robust than clustering the samples directly.

Usage

```
eemsom_cluster(som, k, method = "complete", seed = NULL)
```

Arguments

som	object of class eemsom, see eemsom
k	number of clusters, see eemsom_kselect for help choosing k
method	"complete" (default, Ejarque-Gonzalez and Butturini 2014) or any other method of hclust, or "kmeans"
seed	seed for the random number generator, only used for method = "kmeans". With NULL (default), the current random number stream of the session is used, so set.seed before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards.

Value

object of class eemsom with the cluster of each unit in the element `clusters`

References

Vesanto, J., Alhoniemi, E. (2000). Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586-600. doi:10.1109/72.846731

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

`eemsom_kselect`, `eemsom_split`

Other clustering: `eemsom_kselect()`, `eemsom_plot_silhouette()`, `eemsom_split()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
som <- eemsom_cluster(som, k = 3)
eemsom_samples(som)
eemsom_plot_map(som, type = "cluster")
```

---

eemsom_codes *Prototype EEMs of a self-organising map*

Description

The codebook vectors of the units are converted to EEMs.

Usage

```
eemsom_codes(som, denormalise = FALSE)
```

Arguments

`som`	object of class eemsom, see `eemsom`
`denormalise`	logical, if TRUE, each prototype is multiplied by the mean normalisation factor of the samples mapped to the unit (the mean of all samples for empty units). The result has the scale of the original data.

Details

For maps of PARAFAC loadings (`eempf_som`), the prototype EEM of a unit is the sum of the PARAFAC components weighted by the prototype loadings.

Value

object of class eemlist, samples are named by the unit (e.g. "V1"), see mat2eem

See Also

Other map results: eemsom_predict(), eemsom_quality(), eemsom_samples(), eemsom_split()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
codes <- eemsom_codes(som)
codes
ggeem(codes)
```

---

eemsom_fri *Fluorescence regional integration*

Description

The fluorescence of each sample is integrated within each region. Following Chen et al. (2003), the volumes are normalised by the area of the region, so regions of different size can be compared. Ejarque-Gonzalez and Butturini (2014) quantified their fluorescence components from the self-organising map in this way.

Usage

```
eemsom_fri(eem_list, regions)
```

Arguments

eem_list	object of class eemlist with the same wavelengths as the self-organising map used for the regions. Usually, these are the original (not normalised) EEMs.
regions	object of class eemsom_regions, see eemsom_regions

Details

The volume of region i is $\Phi_i = \sum I(ex, em)\Delta ex \Delta em$ over all wavelength pairs of the region. The area-normalised volume is $\Phi_i \cdot A/A_i$, with A being the total area of all regions and A_i the area of region i. Missing values in a sample are ignored and the area of the region is reduced accordingly.

Value

data frame with column sample and one column per region (region_1, region_2, ...) containing the area-normalised volumes

References

Chen, W., Westerhoff, P., Leenheer, J. A., Booksh, K. (2003). Fluorescence Excitation-Emission Matrix Regional Integration to Quantify Spectra for Dissolved Organic Matter. Environmental Science & Technology 37(24), 5701-5710. doi:10.1021/es034354c

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other fluorescence regions: `eemsom_regions()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
regions <- eemsom_regions(som, k = 4, dim = c(8, 8), seed = 42)
eemsom_fri(eem_list, regions)
```

---

eemsom_kselect *Help choosing the number of clusters*

Description

The units of a self-organising map are clustered with different numbers of clusters k. Several criteria are calculated for each k.

Usage

```
eemsom_kselect(som, k = 2:9, method = "complete", seed = NULL)
```

Arguments

`som`	object of class eemsom, see `eemsom`
`k`	numeric vector of numbers of clusters to be tested
`method`	clustering method, see `eemsom_cluster`
`seed`	seed for the random number generator, only used for `method = "kmeans"`. With `NULL` (default), the current random number stream of the session is used, so `set.seed` before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards.

Details

- silhouette: mean silhouette width of the units (Rousseeuw 1987), higher is better
- negative: number of units with negative silhouette width, i.e. units that are closer to another cluster, lower is better (Ejarque-Gonzalez and Butturini 2014)
- davies_bouldin: Davies-Bouldin index, lower is better (Inf if two clusters have identical centres)
- wss: within-cluster sum of squares, look for an elbow

Value

data frame with one row per k

References

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53-65. doi:10.1016/0377-0427(87)901257

Davies, D. L., Bouldin, D. W. (1979). A Cluster Separation Measure. IEEE Transactions on Pattern Analysis and Machine Intelligence 1(2), 224-227. doi:10.1109/TPAMI.1979.4766909

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other clustering: `eemsom_cluster()`, `eemsom_plot_silhouette()`, `eemsom_split()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
eemsom_kselect(som, k = 2:6)
```

`eemsom_plot_cplane` *Plot component planes of a self-organising map*

Description

A component plane shows the value of one variable in the prototypes of all units. Similar component planes indicate correlated variables. For maps of EEMs, the variables are wavelength pairs, for maps of PARAFAC loadings (`eempf_som`), the variables are the PARAFAC components.

Usage

```
eemsom_plot_cplane(som, ex = NULL, em = NULL, comps = NULL, scale = "common")
```

Arguments

`som`	object of class eemsom, see `eemsom` or `eempf_som`
`ex`	numeric vector of excitation wavelengths (maps of EEMs only)
`em`	numeric vector of emission wavelengths, same length as `ex`. Each pair of `ex` and `em` gives one component plane.
`comps`	numeric vector or names of PARAFAC components (maps of PARAFAC loadings only). NULL (default) plots all components.
`scale`	`"common"` (default) uses one colour scale for all component planes, so values can be compared between planes. `"each"` scales each component plane from 0 (minimum) to 1 (maximum), which shows patterns of weak peaks or components more clearly.

Details

The closest wavelengths available in the data are used. With `scale = "common"`, values are on the scale of the normalised data.

Ejarque-Gonzalez and Butturini (2014) compared the component planes of all wavelength pairs by their correlation to discriminate fluorescence components, see `eemsom_regions`.

Value

a ggplot object

References

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other plots: `eemsom_plot_map()`, `eemsom_plot_silhouette()`, `eemsom_plot_stability()`, `eemsom_plot_training()`, `ggeem.eemsom()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
# fluorescence peaks B, T, A and C (Coble 1996)
eemsom_plot_cplane(som, ex = c(275, 275, 260, 350), em = c(310, 340, 450, 450))
eemsom_plot_cplane(som, ex = c(275, 275, 260, 350), em = c(310, 340, 450, 450), scale = "each")

data(pf_models)
pfsom <- eempf_som(pf4[[1]], rlen = 100, seed = 42)
eemsom_plot_cplane(pfsom, scale = "each")
```

`eemsom_plot_map` *Plot a self-organising map*

Description

Different properties of the units are shown on the map.

Usage

```
eemsom_plot_map(
  som,
  type = "counts",
  colour = NULL,
  labels = FALSE,
  unit_labels = FALSE
)
```

Arguments

`som`	object of class eemsom, see `eemsom`
`type`	property to be shown: • `"counts"`: number of samples mapped to each unit • `"umatrix"`: unified distance matrix, mean Euclidean distance between the prototype of a unit and the prototypes of its direct neighbours. High values separate groups of similar units. • `"quality"`: mean distance between the samples of a unit and its prototype, see `eemsom_samples` • `"cluster"`: clusters of the units • `"samples"`: samples as points on their units, the units are coloured by cluster if available
`colour`	optional vector with one value per sample (e.g. sampling site from metadata) used to colour the samples with `type = "samples"`
`labels`	logical, whether sample names are shown with `type = "samples"`
`unit_labels`	logical, whether the numbers of the units are shown. They are used by `ggeem.eemsom`, `eemsom_codes` and `eemsom_samples`.

Details

Neighbours are defined like in `unit.distances`: six units on hexagonal maps, eight units including the diagonals on rectangular maps.

With `type = "samples"`, samples of the same unit are arranged in a circle around the centre of the unit.

Ejarque-Gonzalez and Butturini (2014) used the U-matrix to find regions of similar units and projected the samples, coloured by sampling site and hydrological conditions, onto the map to explore the patterns among samples.

Value

a ggplot object

References

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other plots: `eemsom_plot_cplane()`, `eemsom_plot_silhouette()`, `eemsom_plot_stability()`, `eemsom_plot_training()`, `ggeem.eemsom()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
eemsom_plot_map(som, unit_labels = TRUE)
eemsom_plot_map(som, type = "umatrix")
eemsom_plot_map(som, type = "samples", colour = substr(eem_names(eem_list), 1, 2), labels = TRUE)
```

---

eemsom_plot_silhouette

Plot silhouette widths of clustered units

Description

The silhouette width of each unit shows how well it fits into its cluster. Values close to 1 indicate a good fit, negative values indicate that the unit is closer to another cluster. Ejarque-Gonzalez and Butturini (2014) chose the number of fluorescence components by the mean silhouette width and the number of negative silhouettes.

Usage

```
eemsom_plot_silhouette(x)
```

Arguments

x	object of class eemsom with clusters (see `eemsom_cluster`) or of class eemsom_regions (see `eemsom_regions`)

Value

a ggplot object

References

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53-65. doi:10.1016/0377-0427(87)901257

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other clustering: `eemsom_cluster()`, `eemsom_kselect()`, `eemsom_split()`

Other plots: `eemsom_plot_cplane()`, `eemsom_plot_map()`, `eemsom_plot_stability()`, `eemsom_plot_training()`, `ggeem.eemsom()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
som <- eemsom_cluster(som, k = 3)
eemsom_plot_silhouette(som)
```

---

eemsom_plot_stability *Plot the neighbourhood stability of a self-organising map*

Description

A histogram of the neighbourhood stability of all pairs of samples is compared to the distribution expected on a randomly organised map (de Bodt et al. 2002), as in Figure 4B of Ejarque-Gonzalez and Butturini (2014).

Usage

```
eemsom_plot_stability(x)
```

Arguments

x	object of class eemsom_stability, see `eemsom_stability`

Value

a ggplot object

References

de Bodt, E., Cottrell, M., Verleysen, M. (2002). Statistical tools to assess the reliability of self-organizing maps. Neural Networks 15(8-9), 967-978. doi:10.1016/S08936080(02)000710

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

Other stability: `eemsom_stability()`

Other plots: `eemsom_plot_cplane()`, `eemsom_plot_map()`, `eemsom_plot_silhouette()`, `eemsom_plot_training()`, `ggeem.eemsom()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, dim = c(3, 3), rlen = 20, seed = 42)
stab <- eemsom_stability(eem_list, som, nrep = 3, seed = 42)
eemsom_plot_stability(stab)
```

---

eemsom_plot_training *Plot the training progress of a self-organising map*

Description

The mean distance between samples and their closest unit during the training is plotted. The curve should flatten at the end of the training, otherwise `rlen` might be increased.

Usage

```
eemsom_plot_training(som)
```

Arguments

`som`	object of class eemsom, see `eemsom`

Value

a ggplot object

See Also

Other plots: `eemsom_plot_cplane()`, `eemsom_plot_map()`, `eemsom_plot_silhouette()`, `eemsom_plot_stability()`, `ggeem.eemsom()`, `ggeem.eemsom_regions()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
eemsom_plot_training(som)
```

`eemsom_predict` *Map new samples to a self-organising map*

Description

New EEMs are assigned to their best matching unit. For maps of EEMs (`eemsom`), they are normalised like the training data. For maps of PARAFAC loadings (`eempf_som`), their loadings are calculated first with the fixed emission and excitation loadings of the PARAFAC model (`A_missing`).

Usage

```
eemsom_predict(som, eem_list, cores = 1, ...)
```

Arguments

`som`	object of class eemsom, see `eemsom` or `eempf_som`
`eem_list`	object of class eemlist with the same excitation and emission wavelengths as the training data or the PARAFAC model
`cores`	number of cores used by `A_missing` (maps of PARAFAC loadings only)
`...`	additional arguments passed on to `A_missing` and `eem_parafac` (maps of PARAFAC loadings only), e.g. `maxit`. By default, `nstart = 1` is used because the loadings with fixed B and C modes do not depend on the start.

Details

Maps of EEMs: only the wavelength pairs used in the map are considered. Missing values in the new samples are ignored when calculating the distances, which are scaled up to the number of all wavelength pairs as in kohonen (see `eemsom_samples`).

Maps of PARAFAC loadings: the new EEMs need the same preparation as the EEMs of the PARAFAC model (e.g. scatter removal). Their Fmax loadings are scaled like in `eempf_som` (relative or absolute) and the distances refer to these loadings.

Value

data frame with columns sample, unit, x, y, distance, hits, umatrix and cluster like `eemsom_samples`. hits is the number of training samples mapped to the unit. A distance much higher than the distances of the training samples indicates a sample that is not represented by the map.

See Also

Other map results: `eemsom_codes()`, `eemsom_quality()`, `eemsom_samples()`, `eemsom_split()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
som <- eemsom_cluster(som, k = 3)
# usually, new samples are mapped. Here, the training samples are used.
eemsom_predict(som, eem_list)

# map of PARAFAC loadings: the loadings of the new samples are calculated first
data(pf_models)
data(eem_list_outliers)
pfsom <- eempf_som(pf4[[1]], rlen = 100, seed = 42)
new_samples <- eem_list_outliers |>
  eem_rem_scat(remove_scatter = c(TRUE, TRUE, TRUE, TRUE),
               remove_scatter_width = c(15, 15, 18, 19)) |>
  eem_interp(type = 1, extend = FALSE, cores = 1)
eemsom_predict(pfsom, new_samples)
```

`eemsom_quality` *Quality measures of a self-organising map*

Description

Quality measures of a self-organising map

Usage

```
eemsom_quality(som)
```

Arguments

`som`	object of class eemsom, see `eemsom`

Details

- quantisation_error: mean Euclidean distance between samples and their prototypes, see `eemsom_samples` for the handling of missing values
- topographic_error: share of samples whose best and second best matching units are not neighbours (six neighbours on hexagonal maps, eight including the diagonals on rectangular maps, see `unit.distances`)
- explained_variance: share of the total sum of squares of the (normalised) data explained by the prototypes

Value

data frame with one row

References

Kohonen, T. (2013). Essentials of the self-organizing map. Neural Networks 37, 52-65. doi:10.1016/j.neunet.2012.09.018

See Also

Other map results: `eemsom_codes()`, `eemsom_predict()`, `eemsom_samples()`, `eemsom_split()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
eemsom_quality(som)
```

---

eemsom_regions *Find fluorescence regions from correlated component planes*

Description

Wavelength pairs with similar behaviour across the units of a self-organising map are grouped into regions. Each region is assumed to represent a fluorescence component (Ejarque-Gonzalez and Butturini 2014). The component planes are arranged on a second self-organising map ("R-mode") and its units are clustered hierarchically.

Usage

```
eemsom_regions(
  som,
  k = 2:9,
  method = "planes",
  dim = "auto",
  cluster_method = "complete",
  seed = NULL,
  ...
)

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

Arguments

`som`	object of class eemsom, see `eemsom`
`k`	number of regions (a single number) or candidates (a vector, default 2 to 9) from which the number with the highest mean silhouette width is chosen. NULL is the same as `2:9`.
`method`	`"planes"` (default) or `"cormat"`, see details

dim	size of the R-mode map, either "auto" (default) or a numeric vector c(xdim, ydim), see eemsom
cluster_method	clustering method of the R-mode units, see eemsom_cluster
seed	seed for the random number generator. With NULL (default), the current random number stream of the session is used, so set.seed before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards.
...	additional parameters that are passed on to som. The default number of iterations is rlen = 100.
x	object of class eemsom_regions

Details

Following Barreto-Sanz and Perez-Uribe (2007), Ejarque-Gonzalez and Butturini (2014) calculated a self-organising map from the matrix of Pearson's correlations between all component planes (method = "cormat"). This matrix has as many rows and columns as wavelength pairs and the calculation gets slow with a high resolution of the EEMs (minutes with a few thousand wavelength pairs).

With method = "planes", each component plane is centred and scaled to length 1. The squared Euclidean distance of two such vectors equals $2(1 - r)$, where r is their correlation. The R-mode map is calculated from these vectors directly, which is much faster and groups wavelength pairs by their correlation as well.

Both methods group wavelength pairs by the correlation of their component planes, but the R-mode maps are trained on different vectors. The resulting regions are similar but not identical and the region numbers can differ.

Wavelength pairs with constant component planes have no correlation and get no region (NA), as well as wavelength pairs that were not used in the map.

Value

object of class eemsom_regions, a list containing

- regions: data frame with columns ex, em and region
- k: number of regions
- kselect: data frame with criteria for the candidates (see eemsom_kselect), NULL if k was a single number
- rsom: R-mode self-organising map of class kohonen
- unit_regions: region of each unit of the R-mode map
- method, cluster_method, seed: settings used. Without a given seed, a seed is drawn from the random number stream of the session and used for the map and the clustering.

References

Barreto-Sanz, M. A., Perez-Uribe, A. (2007). Improving the Correlation Hunting in a Large Quantity of SOM Component Planes. In: Artificial Neural Networks - ICANN 2007, Lecture Notes in Computer Science 4669, 379-388. doi:10.1007/9783540746959_39

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

eemsom_fri

Other fluorescence regions: eemsom_fri(), ggeem.eemsom_regions()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
regions <- eemsom_regions(som, dim = c(8, 8), seed = 42)
regions
regions$kselect
ggeem(regions)
# fixed number of regions
eemsom_regions(som, k = 3, dim = c(8, 8), seed = 42)
```

---

eemsom_samples *Assignment of samples to units of a self-organising map*

Description

Assignment of samples to units of a self-organising map

Usage

```
eemsom_samples(som)
```

Arguments

som	object of class eemsom, see eemsom

Details

With missing values (na = "keep"), the distances are calculated from the available wavelength pairs of a sample and scaled up to the number of all wavelength pairs, as in kohonen.

The columns hits and umatrix help to find unusual samples (outliers): such samples often occupy a unit alone (hits = 1) that is separated from its neighbours by a high U-matrix value. Their distance to the prototype can be small though, because the prototype adapts to them. The leave-one-out distance calculated by eemsom_stability with type = "loo" is a more reliable indicator.

Value

data frame with columns sample, unit (number of the best matching unit), x and y (position of the unit on the map), distance (Euclidean distance between sample and prototype), hits (number of samples mapped to the unit), umatrix (mean distance between the prototype of the unit and the prototypes of its neighbours, see eemsom_plot_map) and cluster (NA if no clustering was done)

See Also

Other map results: eemsom_codes(), eemsom_predict(), eemsom_quality(), eemsom_split()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
eemsom_samples(som)
```

---

eemsom_split *Split an eemlist according to the clusters or units of a self-organising map*

Description

The samples are divided into groups that can be analysed separately, e.g. by eem_parafac.

Usage

```
eemsom_split(eem_list, som, by = "cluster", min_samples = 20)
```

Arguments

eem_list	object of class eemlist, containing samples used to calculate the self-organising map. Usually, these are the original (not normalised) EEMs.
som	object of class eemsom, see eemsom
by	"cluster" (default) or "unit"
min_samples	minimum number of samples per group. A warning is shown for smaller groups because a PARAFAC model might not be stable.

Details

Samples are matched by their names. Groups without samples are omitted.

Value

a named list of eemlists, names are "cluster_1", "cluster_2", ... or "unit_1", ...

See Also

eemsom_cluster, eem_parafac

Other clustering: eemsom_cluster(), eemsom_kselect(), eemsom_plot_silhouette()

Other map results: eemsom_codes(), eemsom_predict(), eemsom_quality(), eemsom_samples()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
som <- eemsom_cluster(som, k = 3)
eem_groups <- eemsom_split(eem_list, som, min_samples = 3)
lengths(eem_groups)


# separate PARAFAC models for the groups with at least 5 samples
# (with real data sets, groups need more samples, starts and iterations)
eem_groups <- eem_groups[lengths(eem_groups) >= 5]
pf_groups <- lapply(eem_groups, eem_parafac, comps = 2, nstart = 1, maxit = 500, cores = 1)
```

---

eemsom_stability *Stability of a self-organising map*

Description

Self-organising maps are calculated repeatedly from resampled data with the same settings as the original map. The stability of the quantisation and of the neighbourhood relations of the samples is assessed (de Bodt et al. 2002).

Usage

```
eemsom_stability(
  eem_list,
  som,
  type = "bootstrap",
  nrep = 20,
  radius = 1,
  seed = NULL,
  cores = 1,
  verbose = FALSE
)

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

Arguments

eem_list	object of class eemlist, the samples used to calculate som
som	object of class eemsom, see eemsom
type	"bootstrap" (default, de Bodt et al. 2002) or "loo" (leave-one-out, Ejarque-Gonzalez and Butturini 2014)
nrep	number of maps (type = "bootstrap") or number of maps per left-out sample (type = "loo")
radius	two samples are neighbours if their units are not more than radius units apart
seed	seed for the random number generator. With NULL (default), the current random number stream of the session is used, so set.seed before the call gives reproducible results. With a number, the seed is set for this calculation only and the random number state of the session is restored afterwards. Each map gets its own seed drawn from this stream, so the result does not depend on cores (with the default random number generator, see RNGkind).
cores	number of parallel calculations
verbose	print infos, with cores = 1 also the progress
x	object of class eemsom_stability
...	further arguments, not used

Details

For each map, all samples are mapped to their best matching unit, including samples that were not part of the resampled data.

Quantisation: the intra-class sum of squares (SS intra), the sum of the squared distances between samples and their prototypes, is calculated for each map. A low coefficient of variation indicates a stable quantisation.

Neighbourhood: for each pair of samples, the stability is the share of maps in which both samples are neighbours. Stable maps have values close to 0 or 1. On a randomly organised map, two samples are neighbours with probability p (element random), the stability then follows a binomial distribution.

Outliers: with type = "loo", each sample is mapped to the maps calculated without it. Its distance to the best matching unit is high, if no other sample is similar, which corresponds to a large residual of a sample that is not part of a PARAFAC model. The mean distance over nrep maps is returned in the element loo. Samples with distinctly higher values than the others are candidates for outliers, which should be checked further, e.g. by eempf_leverage and eempf_residuals_plot in a PARAFAC analysis.

Run time: nrep maps (type = "bootstrap") or nrep times the number of samples maps (type = "loo") are trained with the settings of som, so the run time is the corresponding multiple of the training time of eemsom. Please see there for ways to reduce the training time and use cores for parallel calculation.

Value

object of class eemsom_stability, a list containing

- quantisation: data frame with columns rep, ss_intra and left_out (only for `type = "loo"`)
- neighbourhood: data frame with columns sample1, sample2 and stability
- random: probability of two samples being neighbours on a random map
- loo: only for `type = "loo"`, data frame with columns sample and distance (mean distance of each sample to its best matching unit on the maps calculated without it)
- nmaps, type, radius: number of maps and settings

References

de Bodt, E., Cottrell, M., Verleysen, M. (2002). Statistical tools to assess the reliability of self-organizing maps. Neural Networks 15(8-9), 967-978. doi:10.1016/S08936080(02)000710

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

`eemsom_plot_stability`

Other stability: `eemsom_plot_stability()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, dim = c(3, 3), rlen = 20, seed = 42)
stab <- eemsom_stability(eem_list, som, nrep = 3, seed = 42)
stab
eemsom_plot_stability(stab)


# in parallel; with real data sets, use more repetitions
som <- eemsom(eem_list, rlen = 100, seed = 42)
stab <- eemsom_stability(eem_list, som, nrep = 4, cores = 2, seed = 42)
eemsom_plot_stability(stab)
```

---

ggeem.eemsom *Plot prototype EEMs of a self-organising map*

Description

The prototype EEMs (codebook vectors) of all units or selected units are plotted. If all units are plotted, the facets are arranged like the units on the map (top row of the map first).

Usage

```
## S3 method for class 'eemsom'
ggeem(data, fill_max = FALSE, units = NULL, denormalise = FALSE, ...)
```

Arguments

data	object of class eemsom, see eemsom
fill_max	sets the maximum fluorescence value for the colour scale, see ggeem
units	optional numeric vector of units to be plotted
denormalise	logical, see eemsom_codes
...	parameters passed on to ggeem, e.g. colpal or contour

Details

The facet titles show the unit and the number of samples mapped to this unit. On a hexagonal map, every second row is shifted by half a unit, which cannot be shown by the facets.

Ejarque-Gonzalez and Butturini (2014) used the prototype EEMs to describe the characteristic fluorescence of the regions of the map.

Value

a ggplot object

References

Ejarque-Gonzalez, E., Butturini, A. (2014). Self-Organising Maps and Correlation Analysis as a Tool to Explore Patterns in Excitation-Emission Matrix Data Sets and to Discriminate Dissolved Organic Matter Fluorescence Components. PLoS ONE 9(6), e99618. doi:10.1371/journal.pone.0099618

See Also

ggeem, eemsom_codes

Other plots: eemsom_plot_cplane(), eemsom_plot_map(), eemsom_plot_silhouette(), eemsom_plot_stability(), eemsom_plot_training(), ggeem.eemsom_regions()

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
ggeem(som)
ggeem(som, units = c(1, 2), contour = TRUE)
```

`ggeem.eemsom_regions` *Plot fluorescence regions*

Description

The regions found by `eemsom_regions` are shown in the excitation-emission space.

Usage

```
## S3 method for class 'eemsom_regions'
ggeem(data, fill_max = FALSE, ...)
```

Arguments

`data`	object of class eemsom_regions
`fill_max`	not used, for consistency with `ggeem`
`...`	further arguments, not used

Value

a ggplot object

See Also

Other fluorescence regions: `eemsom_fri()`, `eemsom_regions()`

Other plots: `eemsom_plot_cplane()`, `eemsom_plot_map()`, `eemsom_plot_silhouette()`, `eemsom_plot_stability()`, `eemsom_plot_training()`, `ggeem.eemsom()`

Examples

```
data(eem_list)

som <- eemsom(eem_list, rlen = 100, seed = 42)
regions <- eemsom_regions(som, dim = c(8, 8), seed = 42)
ggeem(regions)
```

`mat2eem` *Fold a matrix of unfolded EEMs back into an eemlist*

Description

Rows of a matrix created by `eem2mat` or of a matrix with the same columns (e.g. codebook vectors of a self-organising map) are converted to EEMs. Wavelength pairs that were removed before are set to NA.

Usage

```
mat2eem(
  mat,
  ex = attr(mat, "ex"),
  em = attr(mat, "em"),
  keep = attr(mat, "keep"),
  samples = rownames(mat),
  norm_factors = NULL
)
```

Arguments

`mat`	matrix with unfolded EEMs in rows
`ex`	excitation wavelengths of the EEMs
`em`	emission wavelengths of the EEMs
`keep`	logical vector of length `length(ex) * length(em)` stating which wavelength pairs are columns of `mat`
`samples`	sample names for the EEMs
`norm_factors`	optional numeric vector, rows of `mat` are multiplied by these factors to reverse a normalisation. NULL (default) leaves the values unchanged.

Value

object of class eemlist. The EEMs have no file name and the attributes used by eemR (e.g. `is_blank_corrected`) are NA.

See Also

`eem2mat`

Other self-organising maps: `eem2mat()`, `eempf_som()`, `eemsom()`

Examples

```
data(eem_list)

m <- eem2mat(eem_list)
eem_list2 <- mat2eem(m)                                          # normalised EEMs
eem_list3 <- mat2eem(m, norm_factors = attr(m, "norm_factors")) # original values
eem_list2
ggeem(eem_list2)
```

Index

∗ **clustering**
  eemsom_cluster, 8
  eemsom_kselect, 11
  eemsom_plot_silhouette, 15
  eemsom_split, 23
∗ **fluorescence regions**
  eemsom_fri, 10
  eemsom_regions, 20
  ggeem.eemsom_regions, 28
∗ **map results**
  eemsom_codes, 9
  eemsom_predict, 18
  eemsom_quality, 19
  eemsom_samples, 22
  eemsom_split, 23
∗ **plots**
  eemsom_plot_cplane, 12
  eemsom_plot_map, 14
  eemsom_plot_silhouette, 15
  eemsom_plot_stability, 16
  eemsom_plot_training, 17
  ggeem.eemsom, 26
  ggeem.eemsom_regions, 28
∗ **self-organising maps**
  eem2mat, 2
  eempf_som, 4
  eemsom, 5
  mat2eem, 29
∗ **stability**
  eemsom_plot_stability, 16
  eemsom_stability, 24

A_missing, *18*

eem2array, *3*
eem2mat, 2, *5, 6*, *29*
eem2mat(), *5*, *8*, *29*
eem_exclude, *7*
eem_parafac, *4, 5*, *18*, *23, 24*
eem_range, *7*
eempf_leverage, *25*
eempf_rescaleBC, *5*
eempf_residuals_plot, *25*
eempf_som, 4, *9*, *12, 13*, *18*
eempf_som(), *3*, *8*, *29*
eemsom, *2*, *4, 5*, 5, *8, 9*, *11*, *13, 14*, *17–23*, *25*, *27*
eemsom(), *3*, *5*, *29*
eemsom_cluster, *4*, 8, *11*, *15*, *21*, *24*
eemsom_cluster(), *12*, *16*, *24*
eemsom_codes, *4*, *8*, 9, *14*, *27*
eemsom_codes(), *18*, *20*, *23, 24*
eemsom_fri, 10, *22*
eemsom_fri(), *22*, *28*
eemsom_kselect, *8, 9*, 11, *21*
eemsom_kselect(), *9*, *16*, *24*
eemsom_plot_cplane, *4*, 12
eemsom_plot_cplane(), *15–17*, *27, 28*
eemsom_plot_map, 14, *23*
eemsom_plot_map(), *13*, *16, 17*, *27, 28*
eemsom_plot_silhouette, 15
eemsom_plot_silhouette(), *9*, *12, 13*, *15*, *17*, *24*, *27, 28*
eemsom_plot_stability, 16, *26*
eemsom_plot_stability(), *13*, *15–17*, *26–28*
eemsom_plot_training, 17
eemsom_plot_training(), *13*, *15–17*, *27, 28*
eemsom_predict, *4*, 18
eemsom_predict(), *10*, *20*, *23, 24*
eemsom_quality, *8*, 19
eemsom_quality(), *10*, *18*, *23, 24*
eemsom_regions, *4*, *10*, *13*, *15*, 20, *28*
eemsom_regions(), *11*, *28*
eemsom_samples, *4*, *6*, *8*, *14*, *18, 19*, 22
eemsom_samples(), *10*, *18*, *20*, *24*
eemsom_split, *4*, *9*, 23
eemsom_split(), *9, 10*, *12*, *16*, *18*, *20*, *23*
eemsom_stability, *4*, *7*, *16*, *22*, 24

eemsom_stability(), *17*

ggeem, *4*, *27, 28*
ggeem.eemsom, *14*, 26
ggeem.eemsom(), *13*, *15–17*, *28*
ggeem.eemsom_regions, 28
ggeem.eemsom_regions(), *11*, *13*, *15–17*, *22*, *27*

hclust, *8*

mat2eem, *2*, *3*, *10*, 29
mat2eem(), *3*, *5*, *8*

norm_array, *2*

print.eempf_som *(*eempf_som*)*, 4
print.eemsom *(*eemsom*)*, 5
print.eemsom_regions *(*eemsom_regions*)*, 20
print.eemsom_stability *(*eemsom_stability*)*, 24

RNGkind, *25*

set.seed, *4*, *6–8*, *11*, *21*, *25*
som, *4–8*, *21*

unit.distances, *14*, *19*