sMapOverlay

November 25, 2014

sMapOverlay	*Function to overlay additional data onto the trained map for viewing the distribution of that additional data*

Description

sMapOverlay is supposed to overlay additional data onto the trained map for viewing the distribution of that additional data. It returns an object of class "sMap". It is realised by first estimating the hit histogram weighted by the neighborhood kernel, and then calculating the distribution of the additional data over the map (similarly weighted by the neighborhood kernel). The final overlaid distribution of additional data is normalised by the hit histogram.

Usage

```
sMapOverlay(sMap, data, additional)
```

Arguments

sMap	an object of class "sMap"
data	a data frame or matrix of input data
additional	a numeric vector or numeric matrix used to overlay onto the trained map. It must have the length (if being vector) or row number (if matrix) being equal to the number of rows in input data

Value

an object of class "sMap", a list with following components:

- nHex: the total number of hexagons/rectanges in the grid
- xdim: x-dimension of the grid
- ydim: y-dimension of the grid
- lattice: the grid lattice
- shape: the grid shape
- coord: a matrix of nHex x 2, with rows corresponding to the coordinates of all hexagons/rectangles in the 2D map grid
- init: an initialisation method

- neighKernel: the training neighborhood kernel
- codebook: a codebook matrix of nHex x ncol(additional), with rows corresponding to overlaid vectors
- hits: a vector of nHex, each element meaning that a hexagon/rectangle contains the number of input data vectors being hit wherein
- mqe: the mean quantization error for the "best" BMH
- call: the call that produced this result

Note

Weighting by neighbor kernel is to avoid rigid overlaying by only focusing on the best-matching map nodes as there may exist several closest best-matching nodes for an input data vector.

See Also

sPipeline, sBMH, sHexDist, visHexMulComp

Examples

```
# 1) generate an iid normal random matrix of 100x10
data <- matrix( rnorm(100*10,mean=0,sd=1), nrow=100, ncol=10)
colnames(data) <- paste(rep(S,10), seq(1:10), sep="")

# 2) get trained using by default setup
sMap <- sPipeline(data=data)

# 3) overlay additional data onto the trained map
# here using the first two columns of the input "data" as "additional"
# codebook in "sOverlay" is the same as the first two columns of codebook in "sMap"
sOverlay <- sMapOverlay(sMap=sMap, data=data, additional=data[,1:2])

# 4) viewing the distribution of that additional data
visHexMulComp(sOverlay)
```