Package 'LSC'

February 19, 2015

Type Package

Title Local Statistical Complexity - Automatic Pattern Discovery in Spatio-Temporal Data

Version 0.1.5

Date 2014-01-20

Author Georg M. Goerg <gmg@stat.cmu.edu>

Maintainer Georg M. Goerg <gmg@stat.cmu.edu>

Description Estimators and visualization for local statistical complexity of (N+1)D fields. In particular for 0, 1 and 2 dimensional space this package provides useful visualization.

License GPL-2

Depends R (>= 2.12.1), LICORS, RColorBrewer, fields, gam, Matrix

URL http://www.gmge.org

Collate 'LICORS2LSC.R' 'LSC-utils.R' 'plot_LSC_1plus1D.R' 'plot_LSC_2plus1D.R' 'states2LSC.R' 'states2probs.R' 'LSC-package.R' 'plot_LSC_0plus1D.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2014-01-24 08:01:49

R topics documented:

LSC-package *Local Statistical Compelxity - Automated Pattern Discovery in Spatio-Temporal Data*

Description

A package to estimate local statistical complexity (LSC), a measure for automated pattern discovery in spatio-temporal data using optimal predictors (see References).

This package is very tightly linked to the **LICORS** package, which can be used to estimate these optimal predictors and state space from data. The **LSC** builds on a known or estimated state space; most estimation is handled by **LICORS** (see `?LICORS`).

There are two ways the state space can be represented: either as a unique state label or as a vector of weights. These two are the principal arguments in the functions of this package:

`weight.matrix` an $N \times K$ matrix, where N are the samples and K are the states. That is, each row contains a vector of length K that adds up to one (the mixture weights).

`states` a vector of length N with entry i being the label $k = 1, \ldots, K$ of PLC i

This is an early release: some function names and arguments might/will (slightly) change in the future, so regularly check with new package updates.

Author(s)

Georg M. Goerg <gmg@stat.cmu.edu>

References

Shalizi, C. R., R. Haslinger, J.-B. Rouquier, K. L. Klinkner, and C. Moore (2006). "Automatic filters for the detection of coherent structure in spatiotemporal systems." Physical Review E 73, 036104

Shalizi, C. R., K. L. Klinkner, and R. Haslinger (2004a). "Quantifying self-organization with optimal predictors." Physical Review Letters 93, 118701.

See Also

The main functions in this package are

- `states2LSC` to estimate LSC from the state space, and
- `LICORS2LSC` which is a wrapper for estimating LSC from a `"LICORS"` class estimate.

Since pattern discovery without visualization is only of very limited use, the `plot.LSC` function shows informative plots for $(1 + 1)D$ and $(2 + 1)D$ systems.

Examples

```
## known predictive state space with a state-vector
data(contCA00)
ll <- states2LSC(states = contCA00$predictive_states - min(contCA00$predictive_states) +
    1)
image2(ll, density = TRUE, legend = FALSE)

# An example using estimates from LICORS
## Not run:
example(LICORS)  # this will give an object 'mod' of class 'LICORS'
image2(LICORS2LSC(mod))

## End(Not run)
```

LICORS2LSC *Estimates LSC from a LICORS estimate*

Description

A wrapper of states2LSC for a 'LICORS' estimate from the **LICORS** package (in particular the output from the mixed_LICORS function). Converts LICORS estimates into an array with LSC.

Usage

```
LICORS2LSC(object, type = c("weights", "argmax"))
```

Arguments

object	an object of class "LICORS"
type	should marginal state probabilities be computed based on the unique state space assignment ("argmax") or using the soft thresholding from the weight matrix ("weights").

Value

An object of class "LSC"

See Also

states2LSC, LSC-utils

Examples

```
## Not run:
# see 2nd example in 'LSC-package'

## End(Not run)
```

LSC-utils *Utilities for LSC class*

Description

The "LSC" class lies at the core of this package as it describes spatio-temporal patterns in the data. It is usually an array with the same spatio-temporal resolution as the original dataset.

plot.LSC plots LSC of $(1+1)D$ and $(2+1)D$ systems.

plot_LSC_1plus1D plots LSC for a (1+1)D field.

plot_LSC_2plus1D plots LSC for a (2+1)D field.

plot_LSC_0plus1D plots LSC for a (0+1)D field, i.e., a time series.

Usage

```
## S3 method for class 'LSC'
plot(x, ...)

plot_LSC_1plus1D(z, col = NULL, lsc.unit = "bits", heights = c(2, 5), widths = c(5,
    2))

plot_LSC_2plus1D(z, type = "temporal", time.frames = NULL, zlim = NULL, heights = NULL,
    lsc.unit = "bits", data = NULL, col = NULL)

plot_LSC_0plus1D(z, col = NULL, lsc.unit = "bits", ...)
```

Arguments

x	an object of class "LSC"
...	optional arguments passed to plot_LSC_2plus1D or plot_LSC_1plus1D.
widths	passed to layout for dividing the plotting region horizontally. A vector of length 2: image (left) & temporal average (right)
z	an object of class "LSC"
type	a "temporal" or a "spatial" summary plot of LSC
time.frames	a vector of length ≤ 6 to indicate what frames should be displayed (only for type = "temporal"). If NULL (default) then it chooses them automatically based on valleys and peaks in the spatial average LSC.
zlim	minimum and maximum z values for which colors should be plotted, defaulting to the range of the finite values of z.
lsc.unit	character string (default: "bits") to write next to the color legend
col	colors: either a string decribing a pallette from the RColorBrewer package (see also http://colorbrewer2.org/), or a list of colors (see image for suggestions).
data	(optional) original data to compare to LSC (relevant only for type = "spatial")
heights	passed to layout for dividing the plotting region vertically. If data = NULL a vector of length 2; otherwise a vector of length 3.

See Also

plot.mixed_LICORS, plot_LSC_2plus1D, plot_LSC_1plus1D

Examples

```
## Not run:
data(contCA00)

temp_lsc <- states2LSC(states = contCA00$predictive_states -
    min(contCA00$predictive_states) + 1)
class(temp_lsc) <- c("LSC", "LSC_1plus1D")
plot_LSC_1plus1D(temp_lsc)

## End(Not run)
## Not run:
data(contCA00)

temp_lsc <- states2LSC(states = contCA00$predictive_states -
    min(contCA00$predictive_states) + 1)
temp_lsc_3D <- array(temp_lsc, dim = c(25, 20, 40))
class(temp_lsc_3D) <- c("LSC", "LSC_2plus1D")
plot_LSC_2plus1D(temp_lsc_3D, type = "temporal")
plot_LSC_2plus1D(temp_lsc_3D, type = "spatial")

## End(Not run)
state.sim <- rpois(100, 1)

lsc.est <- states2LSC(states = state.sim)
class(lsc.est) <- c("LSC", "LSC_0plus1D")
plot_LSC_0plus1D(lsc.est)

weights.sim <- matrix(runif(1000, 0, 1), ncol = 10)
weights.sim <- normalize(weights.sim)
lsc.est <- states2LSC(weight.matrix = weights.sim)
plot_LSC_0plus1D(lsc.est)
```

`states2LSC` *Estimate local statistical complexity (LSC) from states*

Description

Converts states (either as a size N array of labels or an $N \times K$ weight matrix) into N local statistical complexities (LSC) per state.

If `states` is a matrix - representing the state space in the dimensions of the original data - then the LSC output will be formatted automatically to an array of the same shape and dimension.

Usage

```
states2LSC(weight.matrix = NULL, states = NULL, base = 2, type = c("MLE"))
```

Arguments

weight.matrix	$N \times K$ weight matrix
states	array of size N with entry i being the label $k = 1, \ldots, K$ of PLC i.
base	logarithm base for complexity (entropy). Default: base = 2 (thus 'bits').
type	estimation type for the probabilities: "MLE"

See Also

states2probs

---

states2probs *Convert states to vector of probabilities of any given state*

Description

Converts states (either as a size N array of labels or an $N \times K$ weight matrix) into N probabilities per state.

If states is a matrix - representing the state space in the dimensions of the original data - then the probabilities will be formatted automatically to an array of the same shape and dimension.

Usage

```
states2probs(weight.matrix = NULL, states = NULL, type = "MLE")
```

Arguments

states	array of size N with entry i being the label $k = 1, \ldots, K$ of PLC i
weight.matrix	$N \times K$ weight matrix
type	estimation type for the probabilities: c("MLE")

See Also

weight_matrix2states

Examples

```
state.sim <- sample.int(5, 100, replace = TRUE)
prob.state <- states2probs(states = state.sim)
layout(matrix(1:2, ncol = 2))
plot(state.sim, xlab = "", ylab = "state")
plot(prob.state, xlab = "", ylab = "probability")
plot(state.sim, prob.state, xlab = "state", ylab = "probability", type = "h")
```

Index