Smdata Quantification Toolbox: Documentation

1. ChangeLog

Sun Aug 16: `osporns@indiana.edu, max.lungarella@aist.go.jp`

- Released complexity toolbox v0.005 (functions: calcI_det.m, calcC_det.m, smdata_cov.m, smdata_corr.m, smdata_H256.m, smdata_jointH256.m, smdata_MIt.m, smdata_MIcov.m)

Sun Aug 30: `max.lungarella@aist.go.jp`

- Draft documentation
- Released complexity toolbox v.0.01 (added functions: smdata_corr.m, entropy.m, smdata_Hn.m, smdata_roulstonH256.m, smdata_jointHn.m, smdata_MItn.m, smdata_roulstonMIt.m, ksdensity2d.m, isomap.m, isomapii.m, L2_distance.m)

Tue Sep 16: `max.lungarella@aist.go.jp`

- Updated documentation
- Released complexity toolbox v.0.02 (added functions: pca_cov.m, pca_svd.m, sortem.m)

Tue Sep 30: `osporns@indiana.edu, max.lungarella@aist.go.jp`

- Changed title of package and of documention
- Released complexity toolbox v.0.03 (added functions: discretize.m, normalize.m; removed functions: smdata_H256.m, smdata_jointH256.m, smdata_roulstonH256.m)

Sat Oct 11: `max.lungarella@aist.go.jp`

- Fixed a bug in `matscat_plot.m`
- Fixed a bug in `test_ksdensity.m`

- `matscat_plot.m` is also called a Hinton diagram. Added functions `hinton.m` and `test_hinton.m`. The former was adapted from `hintonw.m` from the neural network toolbox.

Tue Apr 6: `maxl@isi.imi.i.u-tokyo.ac.jp`

- Released complexity toolbox v.0.05 (what happened to v.0.04?). Added functions: ksdensity1d.m, gsi.m, test_gsi.m

Mon May 31: `maxl@isi.imi.i.u-tokyo.ac.jp`

- Released complexity toolbox v.0.06. Many unused functions were purged, some functions were integrated with other ones.

2. Short Introduction

3. Overview of the Functions

Function	
smd_acf.m	Auto-correlation
smd_H.m	Entropy
smd_Himages.m	Entropy
smd_jointH.m	Joint entropy
smd_covcor.m	Covariance and correlation
smd_corrt.m	Correlation
smd_ksdens1d.m	Histogram
smd_ksdens2d.m	Histogram
smd_MIt.m	Mutual information
smd_MIcov.m	Mutual information
smd_calcI_det.m	Integration of a system
smd_calcC_det.m	Complexity of a system
isomap.m	Isometric feature mapping
isomapii.m	Isometric feature mapping
smd_pca_cov.m	Principal component analysis
smd_pca_svd.m	Principal component analysis
smd_gsi.m	Linear separability
smd_matscat_plot.m	Visualization
smd_hinton.m	Visualization
smd_discrete.m	Discretization
smd_norm.m	Normalization
sortem.m	Sort of eigenvalues and vectors
L2_distance.m	Euclidean distance
test_discrete.m	Test function
test_gsi.m	Test function
test_hinton.m	Test function
test_ksdensity.m	Test function
test_matscat.m	Test function
test_pca.m	Test function
test_smdata.m	Test function

Table 1:

4. Entropy

4.1 smd_H.m

Synopsis:

`[H_obs,B_star,H_inf,Sigma_H]` = `smd_H(Mstates,nst)`

Input:

`Mstates` = NxM matrix (ensemble of series: columns are realizations of stochastic processes)

`nst` = number of states of input space

Output:

`H_obs` = 1xM (observed entropy)

`B_star` = 1xM (number of states for each column of `Mstates`)

`H_inf` = 1xM vector (corrected estimate of entropy)

`Sigma_H` = 1xM vector (standard deviation from theoretically predicted entropy)

Notes:

(a) `Mstates` must be supplied in discretized format with states ranging from 1 to `nst`. To generate such states refer to `smd_discrete.m`

(b) To obtain a meaningful estimate, a sufficient number of samples of the random variables `Mstates` should be supplied. For example, for 8-bit resolution (`nst` = 256 possible states), N>3x256 = 800 samples, because otherwise there are not enough samples to estimate the histogram.

References:

Roulston (1999)

4.2 smd_Himages.m

Synopsis:

`[Himages]` = `smd_Himages(Mstates,nst)`

Input:

`Mstates` = NxM matrix (each row represents a vector-coded image)

`nst` = number of states of input space

Output:

`Himages` = 1xM (each element represents unbiased entropy estimate of one image)

Notes:

(a) `Mstates` must be supplied in discretized format with states ranging from 1 to `nst`. To generate such states refer to `smd_discrete.m`

(b) To obtain a meaningful estimate, a sufficient number of samples of the random variables `Mstates` should be supplied. For example, for 8-bit resolution (`nst` = 256 possible states), N>3x256 = 800 samples.

4.3 smd_jointH.m

Synopsis:

`[jointH] = smd_jointH(Mstates,units,nst,t)`

Calculates joint entropy for each pair of units in Mstates

Input:

`Mstates` = NxM matrix (realizations of stochastic process)

`units` = 1xL, L≤M (columns of `Mdata` used for calculation)

`nst` = number of states of input space

`t` = time offset used to compute joint entropy

Output:

`jointH` = 1xM vector (time-delayed joint entropy, computed column-wise)

Notes:

`Mstates` must be supplied in discretized format with states ranging from 1 to `nst`.

5. Covariance and Correlation

5.1 smd_covcor.m

Synopsis:

`[COV,COR] = smd_covcor(Mdata,units)`

Calculates the covariance and the correlation matrix

Input:

`Mdata` = NxM matrix (realization of stochastic process)

`units` = 1xL, L≤M (columns of `Mdata` used for calculation)

Output:

`COV` = LxL matrix (covariance)

`COR` = LxL matrix (correlation)

5.2 smd_coort.m

Synopsis:

`[CORRt] = smd_coort(Mdata,units,t,T)`

Calculates the time-delayed correlation matrix

Input:

`Mdata` = NxM matrix (realization of stochastic process)

`units` = 1xL, L≤M (columns of `Mdata` used for calculation)

`t` = time offset used to compute correlation

`T` = time window

Output:

`CORRt` = LxL matrix (time-delayed correlation)

6. Histogram

6.1 smd_ksdens2d.m

Synopsis:

`[D] = smd_ksdens2d(Mdata,gridx1,gridx2,bw)`

Input:

`Mdata` = Nx2 matrix

`gridx1` = Mx1 vector

`gridx2` = Mx1 vector

`bw` = bandwidth (window parameter)

Output:

`D` = Kernel density estimation of 2D histogram using Gaussian windows

References:

Moon, Rajagopalan & Lall (1995)

7. Mutual Information

7.1 smd_MIcov.m

Synopsis:

`[MI] = smd_MIcov(Mstates,units)`

Input:

`Mstates` = NxM matrix (realizations of stochastic process)

`units` = 1xL, L≤M (columns of `Mstates` used for calculation)

Output:

`MI` = LxL matrix (mutual information between all pairs part of units)

7.2 smd_MIt.m

Synopsis:

`[MIt_inf,MIt_obs] = smd_MIt(Mstates,units,nst,t)`

Input:

`Mstates` = NxM matrix (realizations of stochastic process)

`units` = 1xL, L≤M (columns of `Mstates` used for calculation)

`nst` = number of states of input space

`t` = time offset used to compute mutual information

Output:

`MIt_inf` = MxM matrix (corrected time-delay mutual information between all pairs of elements that are part of units)

`MIt_obs` = MxM matrix (observed mutual information)

References:

Roulston (1999); Steuer, Kurths, Daub, Weise & Selbig (2002)

8. Complexity Measures

8.1 smd_calcI_det.m

Synopsis:

`[I] = smd_calcI_det(COV)`

Input:

`COV` = covariance matrix of system X

Output:

`I` = integration

Notes:

Assumption of Gaussianity for X

References:

Olaf?

8.2 smd_calcC_det.m

Synopsis:

`[C] = smd_calcC_det(COV)`

Input:

`COV` = covariance matrix of system X

Output:

`C` = complexity

Notes:

Uses `smd_calcI_det.m`

References:

Olaf?

9. Dimensionality Reduction

9.1 smd_pca_cov.m

Synopsis:

`[Z,U,lambda] = smd_pca_cov(Mdata,A)`

Input:

`Mdata` = NxM matrix (rows: observations, columns: variables)

`A` = indeces of returned principal components

Output:

`Z` = NxA matrix (z-loading)

`U` = MxA matrix (principal components)

`lambda` = Ax1 vector (explained variance)

9.2 smd_pca_svd.m

Synopsis:

`[Z,U,lambda] = smd_pca_svd(Mdata,A)`

Input:

`Mdata` = NxM matrix (rows: observations, columns: variables)

`A` = indeces of returned principal components

Output:

`U` = MxA matrix (principal components)

`lambda` = Ax1 vector (explained variance)

9.3 smd_gsi.m

Synopsis:

`[GSI] = smd_gsi(Mdata,Catg)`

Input:

`Mdata` = NxM matrix (rows: observations, e.g. sensory patterns; columns: variables)

`Catg` = Nx1 vector (category vector, category to which observations or patterns belong)

Output:

`GSI` = Geometric Separability Index

References:

Thornton (1997)

9.4 isomap.m

Synopsis:

`[Y,R,E]=isomap(D,n_fcn,n_size,options)`

Input:

Output:

References:

Tenenbaum, de Silva & Langford (2000)

9.5 isomapii.m

Synopsis:

`[Y,R,E]=isomapii(D,n_fcn,n_size,options)`

Input:

Output:

References:

Tenenbaum, de Silva & Langford (2000)

10. Visualization

10.1 smd_matscat_plot.m

Synopsis:

`smd_matscat_plot(M,Z,f,marker)`

Input:

`M` = Nx2 matrix (sets the position of the patches)

`Z` = Nx1 vector (sets the size of the patches)

`f` = number (scale factor)

`marker` = marker (‘s’:square, ‘o’:circles)

Example:

```
[X,Y]=meshgrid(1:20,1:20);

X=reshape(X,400,1);

Y=reshape(Y,400,1);

M=0:0.01:3.99;

M=M'+0.01;

figure(1); clf;

smd_matscat_plot([X,Y],M,100,'s');
```

11. Miscellania

11.1 smd_discrete.m

Synopsis:

`[Mstates] = smd_discrete(Mdata,nst)`

Discretizes the `Mdata` into `nst` discrete states.

Input:

`Mdata` = NxM matrix (input)

`nst` = scalar (number of states)

Output:

`Mstates` = NxM matrix (ranges from 1 to `nst`)

11.2 smd_norm.m

Synopsis:

`[Mdata_out] = smd_norm(Mdata_in,lo_limit,hi_limit)`

Normalizes `Mdata_in` between the user-supplied limits `lo_limit` and `hi_limit`.

Input:

`Mdata_in` = NxM matrix

`lo_limit` = scalar (lower limit)

`hi_limit` = scalar (upper limit)

Output:

`Mdata_out` = NxM matrix

11.3 sortem.m

Synopsis:

`[n,v] = sortem(nd,nv)`

Input:

Output:

11.4 L2_distance.m

Synopsis:

`[D] = L2_distance(A,B)`

Computes Euclidean distance matrix

Input:

A = NxM matrix

B = NxP matrix

Output:

D = MxP matrix

References

Moon, Y., Rajagopalan, B. & Lall, U. (1995). Estimation of mutual information using kernel density estimators. *Physical Review E*, **52**, 2318–2321.

Roulston, M. (1999). Estimating the errors on measured entropy and mutual information. *Physica D*, **125**, 285–294.

Steuer, R., Kurths, J., Daub, C., Weise, J. & Selbig, J. (2002). The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics*, **18**, 231–240, suppl.2.

Tenenbaum, J., de Silva, V. & Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, **290**, 2319–2323.

Thornton, C. (1997). Separability is a learner's best friend. In *Proc. of 4th Workshop on Neural Computation and Psychology: Connectionists Representations*, 40–47.