information-dynamics-toolkit

Java Information Dynamics Toolkit for studying information-theoretic measures of computation in complex systems

SchreiberTeDemos

Demos to recreate Schreiber's original transfer entropy examples

octave, matlab

Updated Today (moments ago) by joseph.lizier

Demos > Schreiber Transfer Entropy Demos

Schreiber Transfer Entropy Demos

This demonstration set (available from release 1.0) shows how to recreate the examples in Schreiber's original paper introducing transfer entropy:

- T. Schreiber, "Measuring Information Transfer", Physical Review Letters 85(2): 461-464.

Several additional analyses on the same data set are provided as well.

It is written for MATLAB or Octave.

This demonstration set is found at demos/octave/SchreiberTransferEntropyExamples/ in the svn or full distribution.

The examples included are:

1. Tent Map
2. Ulam Map
3. Heart-breath interaction

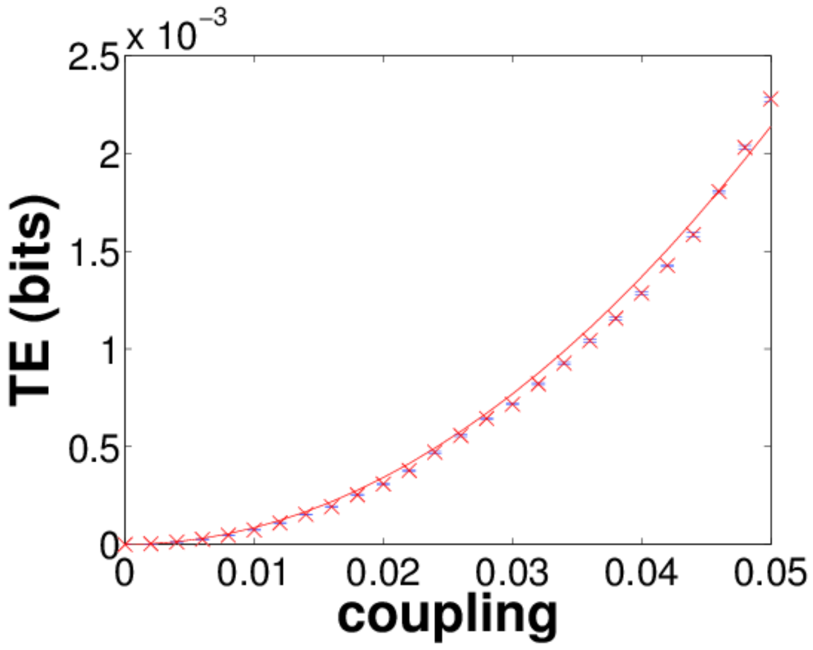
1. Tent Map

The script runTentMap.m recreates Figure 1 from Schreiber's TE paper. It is run simply as follows:

```
teValues = runTentMap();
```

This script runs the Tent Map, with all parameters as described in the paper, then computes the TE from a discretization of the data. The script as it is takes 15 - 30 minutes to run (for all coupling strengths and 10 repeat runs), though you can edit the number of iterates etc to make it run faster (e.g. using 10000 iterates instead of 100000 runs ~10x faster and gives comparable results).

The script finally plots the TE as a function of the coupling parameter of the Tent Map, as shown here:



Note that the line was Schreiber's line of best fit, which may not necessarily be quite the line of best fit from our run.

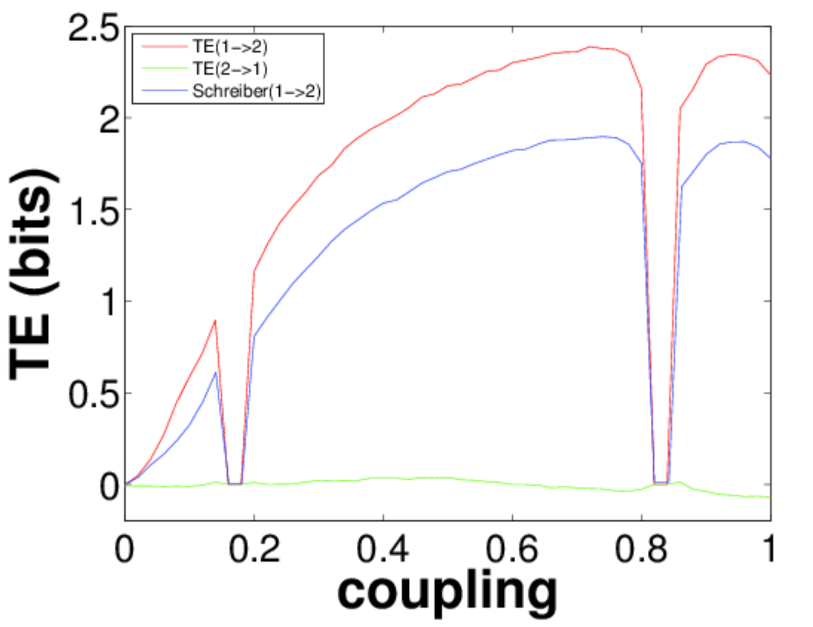
2. Ulam Map

The script runUlamMap.m recreates Figure 2 from Schreiber's TE paper. It is run simply as follows:

```
[teValues1to2, teValues2to1] = runUlamMap();
```

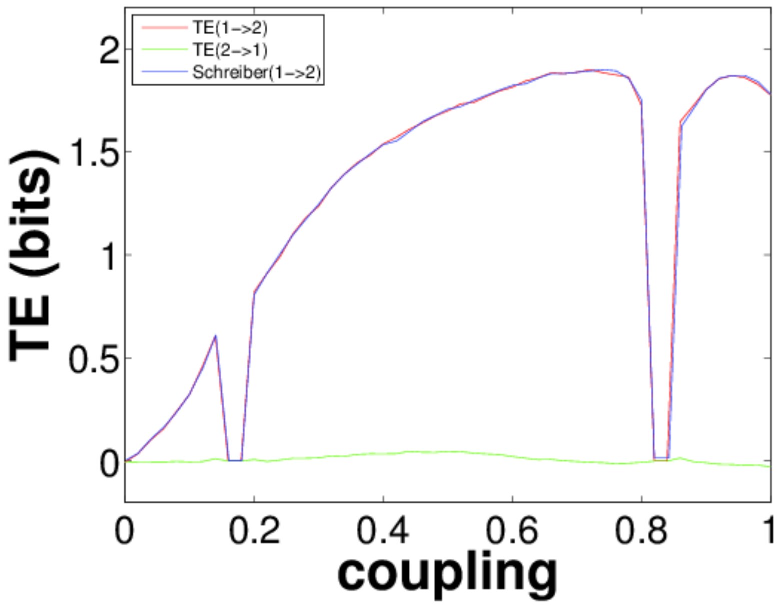
This script runs the Ulam Map, with all parameters as described in the paper (subject to comments below), then computes the TE using a kernel estimator. The script as it is takes 5-10 minutes to run (for all coupling strengths), though you can edit the number of iterates or coupling strength step to make it run faster. The script finally plots the TE as a function of the coupling parameter of the Ulam Map.

Now, if we use the parameters exactly as stated in Schreiber's paper (with kernel width of 0.2) then we get the following result:



where the TE(1->2) is clearly larger than Schreiber's results (which we have recreated here by extraction from his plot, see file SchreiberExample2.txt).

However, if we change the kernel width to 0.3, then we get results which are uncannily similar to Schreiber's:



I have exchanged correspondence with Thomas Schreiber on this, establishing that he in fact added bias correction (as stated in the 2002 Kaiser and Schreiber Physica D paper, though not stated in this paper), though I've found this makes no difference for the 10000 iterates here.

Since we found no other discrepancy (and our kernel estimator is clearly working as per example 3 below), and the match for kernel width of 0.3 is so good, then we conclude that Schreiber seems to have used a kernel width of 0.3 for this example. As such, our script sets the kernel width to 0.3.

3. Heart-breath interaction

The script runHeartBreathRateKernel.m recreates Figure 4 from Schreiber's TE paper. It is run simply as follows:

```
[teHeartToBreath, teBreathToHeart] = runHeartBreathRateKernel();
```

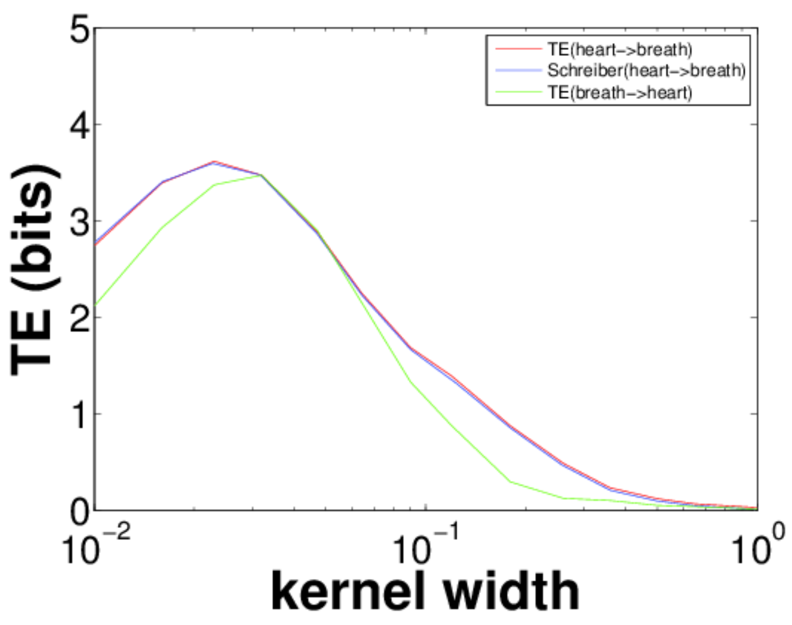
This script loads the heart-breath data, then computes TE using box-kernel estimation for a range of kernel widths.

A number of crucial details were not made explicit in Schreiber's original paper:

1. The individual variables were **normalised** to mean 0 and unit variance. Schreiber doesn't explicitly say this is done for TE, but it is done for the raw data plots in Figure 3. We do this by setting the TE calculator's property `'NORMALISE'` to `true`.
2. **Dynamic correlation exclusion** within 100 time points was used; although Schreiber does not state this for example 3, it is stated for example 2 that "Neighbors closer in time than 100 iterates were excluded from the kernel estimation" and we validate that this is the case here also. We do this by setting the TE calculator's property `'DYN_CORR_EXCL'` to `'100'`.
3. The estimates were **bias corrected** using the approximation log_e(x) ~ digamma(x). Again, this is not stated in the paper, but in correspondence with Schreiber he confirmed that it was done for example 2, and we validate that it is the case here also. We do this by calling the method `computeAverageLocalOfObservationsWithCorrection()` intead of `computeAverageLocalOfObservations()`. Note that we implement bias correction via a separate method rather than by setting a property since we do not wish to make it a proper option here. This is because it is not recommended to use the bias corrected method, since it is not clear how the corrections cancel (see Kaiser and Schreiber, 2002, section 5.2.3.1. It is done properly by the later Kraskov estimator.

The script as it is take a couple of seconds to run.

The script plots the TE as a function of the kernel width, as shown here:



Note the strong match to Schreiber's figure 3 (which we have recreated here by extraction from his plot, see file SchreiberExample3?.txt).