SchreiberTeDemos

Demos to recreate Schreiber's original transfer entropy examples

octave, matlab, examples

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:

1. T. Schreiber, "Measuring Information Transfer", Physical Review Letters **85**(2): 461-464, 2000.

(Or see ref. 1 in #References below). 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
4. Heart-breath interaction - further investigation

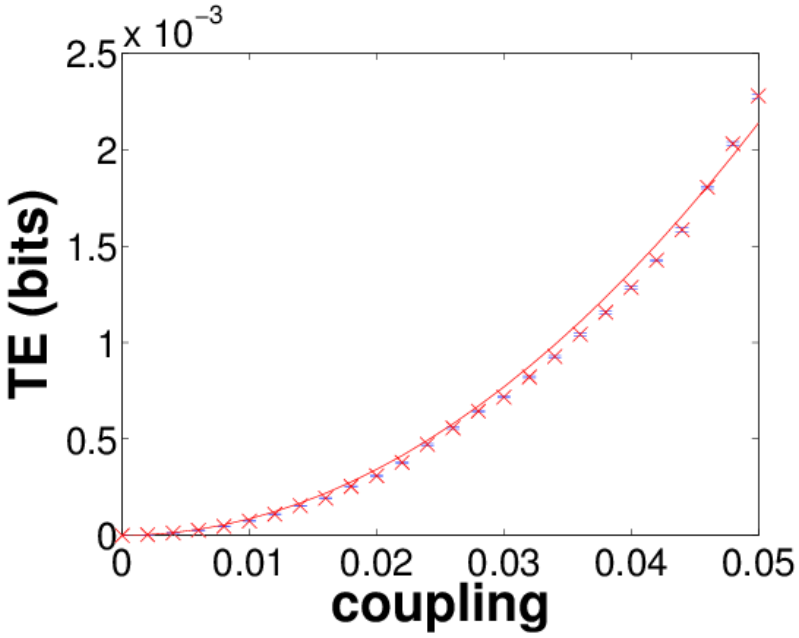
1. Tent Map

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.

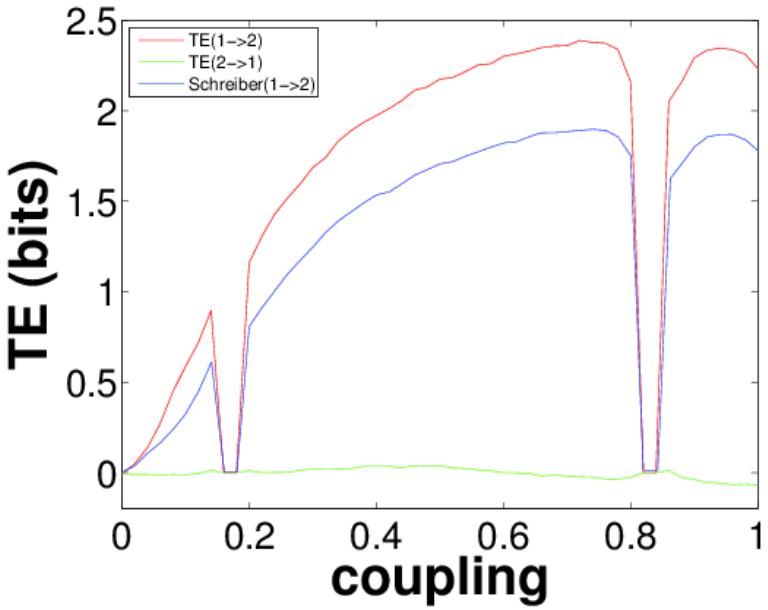
2. Ulam Map

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

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

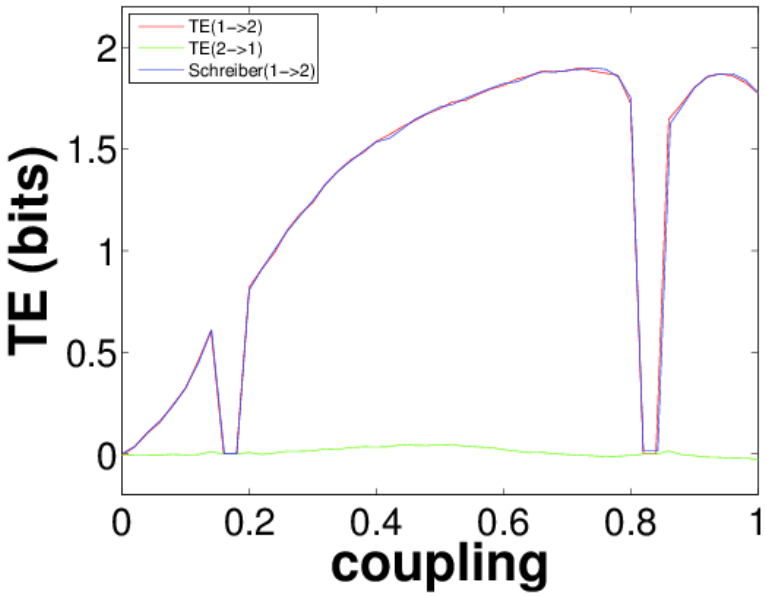
This script runs the Ulam Map, with all parameters as described in the paper (subject to comments below), then computes the TE using a kernel estimator. The script as it is takes 5-10 minutes to run (for all coupling strengths), though you can edit the number of iterates or coupling strength step to 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 Schreiber's later work (ref. 2), though not stated in (ref. 1)), 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();
```

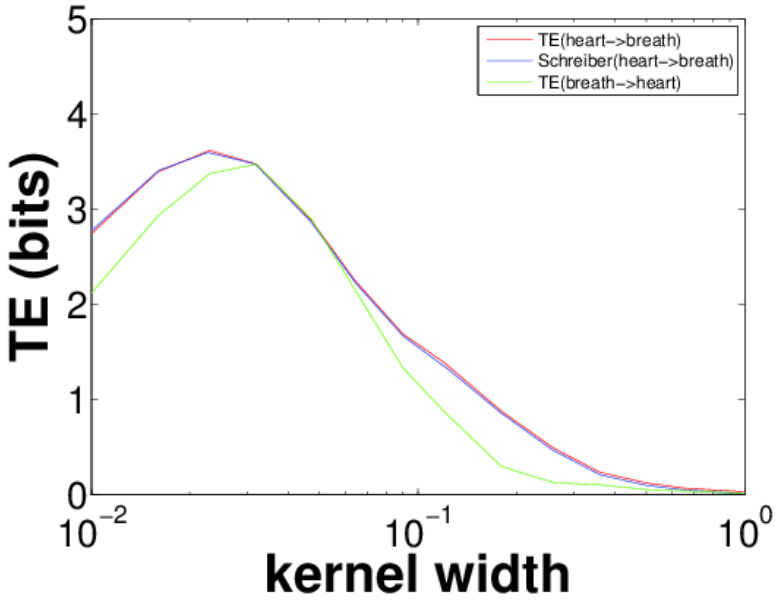
This script loads the heart-breath data (made available via the Santa Fe Institute time series contest held in 1991 (ref. 8) and redistributed with permission from Andreas Weigend), then computes TE using box-kernel estimation for a range of kernel widths.

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) as per (ref. 7). 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 (ref. 2), 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).

4. Heart-breath interaction - further investigation

The original analysis of the heart-breath interaction above leaves several open questions:

1. Were the time-series properly embedded to capture their states and eliminate information storage from the calculation? One can easily use k=2 in the above example (exercise left to reader) to establish that the majority of what is inferred as information transfer by the transfer entropy with k=1 appears to be information storage. Other authors have used k=5 (ref. 3).
2. What is the most appropriate kernel width to use? The results here are highly variable with respect to kernel width, which has been noted in particular in (ref. 2).
3. Was bias eliminated properly using the Grassberger method? We know that it is not clear how the corrections cancel and its use is not recommended (ref. 2).

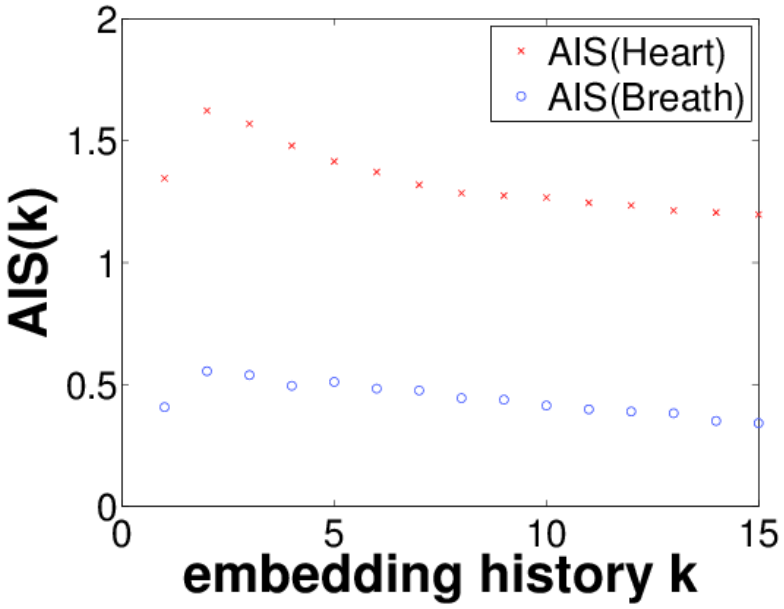
Continuing with a kernel estimator, one could use surrogates to compute the bias (see Example 3 of the Simple Java Examples), and indeed explore the embedding this way, however we are still stuck with the problem of choosing the kernel width.

A better approach is to switch estimators to the newer nearest-neighbour based **Kraskov-Stoegbauer-Grassberger (KSG) estimator** for mutual information (ref. 4), extended to transfer entropy in (ref. 5). That is because this estimator includes automatic bias correction and is very stable to parameter selection (being the number of nearest neighbours). We continue with this type of estimator.

Our first step is to check the embedding dimension. We use the script activeInfoStorageHeartBreathRatesKraskov.m for this purpose. For a given number of K nearest neighbours (here 4), this plots the active information storage (see (ref. 6)) as a function of embedded history length k, e.g.

```
[aisHeart, aisBreath] = activeInfoStorageHeartBreathRatesKraskov(1:15, 4);
```

produces the following plot:

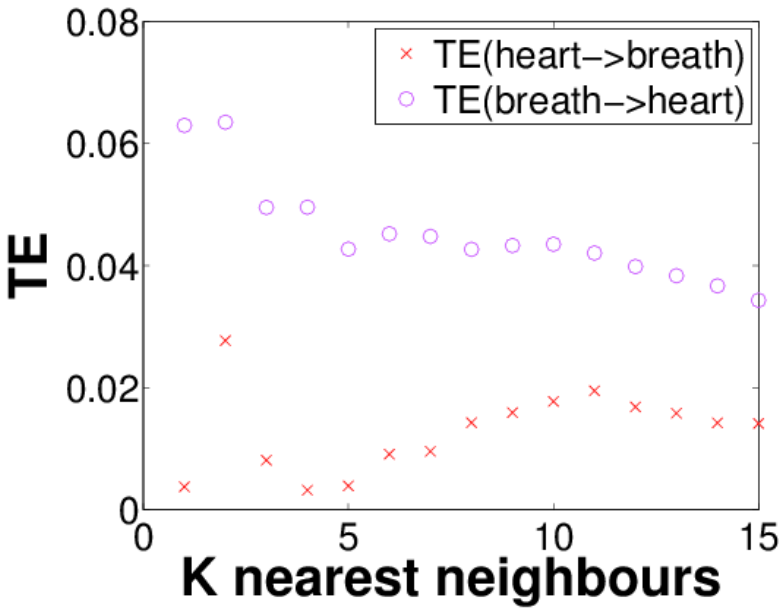


Since the Kraskov estimator is bias corrected (i.e. while a raw MI would normally increase with an increase in dimensionality -- caused by increasing k here -- this one will reduce once we start undersampling), we can use the peak to suggest that an embedded history of k=2 for both heart and breath time-series is appropriate to capture all relevant information from the past without capturing more spurious than relevant information as k increases. This result is stable with the number of nearest neighbours K -- you can check this by changing K=4 above. Another exercise is to supply the third parameter to the script to validate that the AIS values are significantly above the noise floor (left to the reader).

We then turn back to transfer entropy, this time via a Kraskov estimator, using our established embedding lengths of k=l=2 for both series. We use the script runHeartBreathRateKraskov.m for this purpose (this Matlab script is mirrored in Java by HeartBreathRateKraskovRunner.java). For our given embedding lengths, we compute the measure for a number of K nearest neighbours:

```
[teHeartToBreath, teBreathToHeart] = runHeartBreathRateKraskov(2, 2, 1:15);
```

which produces the following plot:



There are a number of important items to observe here:

1. The level of TE was dramatically reduced by proper embedding (to remove information storage) and proper bias correction;
2. The results are above the noise floor for approximately K>=3 for breath->heart and approximately K>=8 for heart->breath. That is, below this scale any apparent TE is a statistical fluctuation simply due to undersampling. One can explore this by using the fourth parameter `numSurrogates` to this script (use 100 to quickly get an estimate of the variance of the noise floor, though for formal results one should use

at least 1000).

3. With an appropriate setting for the number of nearest neighbours K to set the scale avoiding undersampling, we see that the measure is very robust to this parameter.
4. Indeed, the results suggest that breath->heart is the dominant direction for information transfer, though there is of course a complex two-way interaction here.
5. Dynamic correlation exclusion (Theiler window) does make a difference here, but not to the overall conclusions (Exercise: check this by settings the property `DYN_CORR_EXCL` for the calculator).

References

1. T. Schreiber, "Measuring Information Transfer", Physical Review Letters **85**(2): 461-464, 2000.
2. A. Kaiser and T. Schreiber, "Information transfer in continuous processes", Physica D **166**(1-2): 43-62, 2002.
3. N. Ancona, D. Marinazzo, and S. Stramaglia. "Radial basis function approach to nonlinear Granger causality of time series", Physical Review E, **70**: 056221, 2004.
4. A. Kraskov, H. Stoegbauer, and P. Grassberger, "Estimating mutual information", Physical Review E **69**: 066138–066153, 2004.
5. G. Gomez-Herrero, W. Wu, K. Rutanen, M. C. Soriano, G. Pipa, and R. Vicente. "Assessing coupling dynamics from an ensemble of time series", arXiv:1008.0539, 2010.
6. J. T. Lizier, M. Prokopenko, and A. Y. Zomaya, "Local measures of information storage in complex distributed computation", Information Sciences, **208**: 39–54, 2012.
7. P. Grassberger, "Finite sample corrections to entropy and dimension estimates" Physics Letters A, **128**(6-7): 369-373, 1988.
8. D. R. Rigney, A. L. Goldberger, W. Ocasio, Y. Ichimaru, G. B. Moody, and R. Mark, "Multi-Channel physiological data: Description and analysis". In A. S. Weigend and N. A. Gershenfeld, editors, "Time Series Prediction: Forecasting the Future and Understanding the Past", pp. 105–129. Addison-Wesley, Reading, MA, 1993.