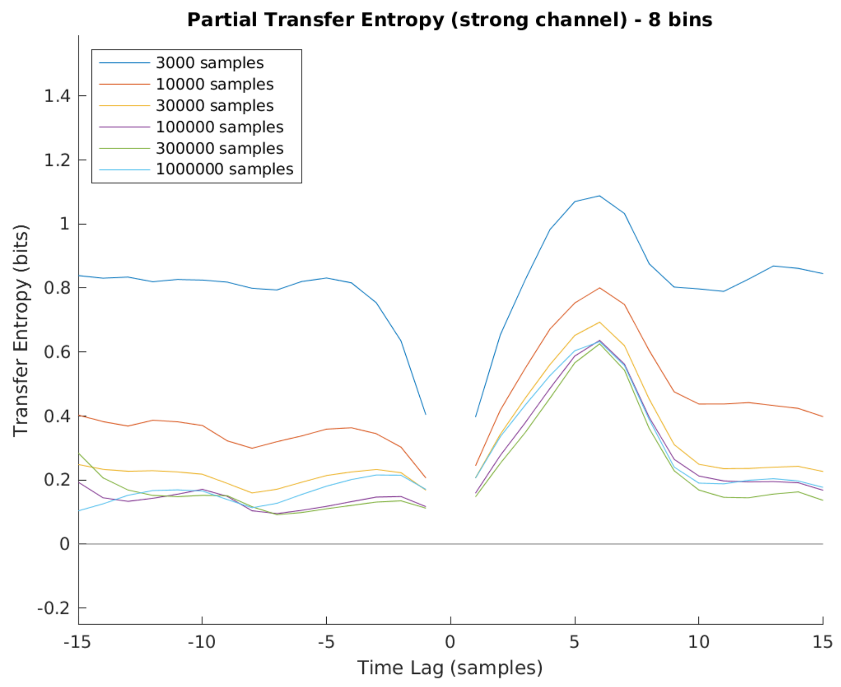
Chris's Entropy Library - User Guide

Written by Christopher Thomas – April 23, 2024.



Contents

Chapter 1

Overview

This library provides a set of functions used for calculating entropy and entropy-related measures on datasets. These were written to deal with neuroscience data (continuous brain wave signals and discrete counts of the numbers spike events seen), but the library should work with any type of data.

Measures computed are:

- **Shannon Entropy** - The amount of "surprise" associated with a given data sample, and the average "surprise" for samples in a data stream. This is the information content of the stream.
- **Conditional Entropy** - The amount learned from samples of variable Y when we already know the value of a related variable X.
- **Mutual Information** - The amount of information shared by samples of related variables Y and X. Measuring either one gets you this information.
- **Transfer Entropy** - The amount of information about the future of variable Y that you learn by knowing the past of variable X, in addition to what you already know from the past of variable Y.

These measures are discussed in detail in Section 2.

A brief description of how to use this library is given in Section 3. For more information, see the sample code in the "`source-code`" folder. This section also provides a brief overview of considerations relating to neuroscience data analysis.

Section 4 describes the quadratic extrapolation algorithm that was used in Palmigiano 2017. This is intended to allow reliable estimates of several entropy measures with fewer data samples than would otherwise be needed (or equivalently, to improve the reliability of these measures using a fixed number of samples).

Chapter 2

Entropy

2.1 Shannon Entropy

Entropy can be thought of as measuring the information content of a stream of data samples. It is defined (per Shannon 1949) as the amount of "surprise" associated with each sample when that sample is considered to be a discrete symbol drawn from a set of possible symbols with some probability distribution. The entropy associated with one symbol is given in Equation 2.1, and the average entropy per symbol associated with a symbol stream is given in Equation 2.2. It can be seen from Equation 2.1 that symbols with lower probability result in more "surprise": rare symbols are more informative than commonly-seen symbols.

$$H(x_k) = \log_2 \left[\frac{1}{P(x_k)} \right] = -\log_2[P(x_k)] \tag{2.1}$$

$$H(X) = -\sum_k P(x_k) \log_2[P(x_k)] \tag{2.2}$$

For discrete-valued data, or data such as text that is readily interpreted as symbols, entropy may be calculated on the raw data values. A histogram of observed symbols is made, and this is used as the probability distribution. For continuous-valued data, histogram bins are typically defined and the bin labels are interpreted as symbols. For data consisting of event arrival times, time bins are typically defined and the number of events within each bin are counted. These counts may be taken as individual symbols or several adjacent time bins may be considered to form a word, which is used as a symbol for entropy calculations. These approaches are discussed in more detail in Section 3.

2.2 Conditional Entropy

Conditional entropy can be thought of as the amount of additional information learned from samples of variable Y when we already know the value of a related variable X. This is illustrated in Figure 2.1 (left). It is defined by Equation 2.3:

$$H(Y|X) = -\sum_{j,k} P(x_j, y_k) \log_2 \left[\frac{P(x_j, y_k)}{P(x_j)} \right] \tag{2.3}$$

For the case of independent variables where $P(x_j, y_k) = P(x_j)P(y_k)$, this reduces to Shannon entropy.

This is generalized to several X variables per Equation 2.4; this case is illustrated in Figure 2.1 (right).

$$H(Y|X_A, X_B) = -\sum_{j,k,m} P(x_{aj}, x_{bk}, y_m) \log_2 \left[\frac{P(x_{aj}, x_{bk}, y_m)}{P(x_{aj}, x_{bk})} \right] \tag{2.4}$$

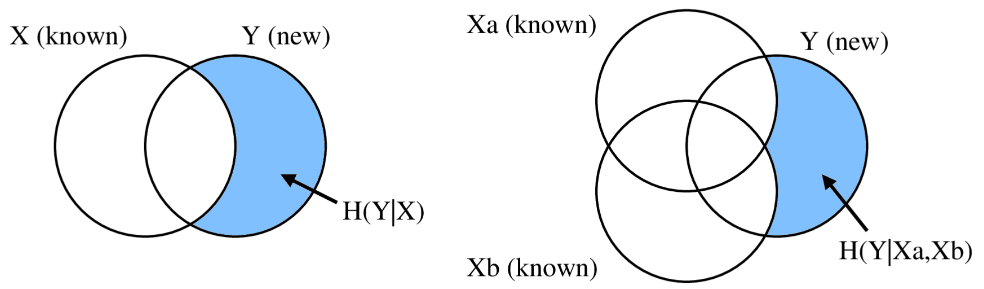


Figure 2.1: Conditional entropy with two signals (left) and three signals (right).

2.3 Mutual Information

Mutual information can be thought of as the amount of information shared between related variables X and Y. Sampling either variable X *or* variable Y reveals the shared information. This is illustrated in Figure 2.2 (left). Mutual information is defined by Equation 2.5:

$$I(X,Y) = \sum_{j,k} P(x_j, y_k) \log_2 \left[\frac{P(x_j, y_k)}{P(x_j)P(y_k)} \right] \tag{2.5}$$

For the case of independent variables where $P(x_j, y_k) = P(x_j)P(y_k)$, this is equal to zero (due to the $log_2(1)$ term).

This is generalized to more than two variables per Equation 2.6; this case is illustrated in Figure 2.2 (right).

$$I(X,Y,Z) = \sum_{j,k,m} P(x_j, y_k, z_m) \log_2 \left[\frac{P(x_j, y_k, z_m)}{P(x_j)P(y_k)P(z_m)} \right] \tag{2.6}$$

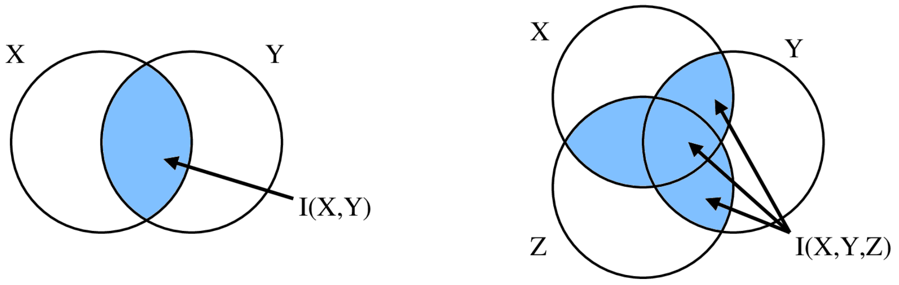


Figure 2.2: Mutual information with two signals (left) and three signals (right).

2.4 Transfer Entropy

Transfer entropy can be thought of as the amount of information about the future of variable Y that you learn by knowing the past of variable X, in addition to what you already know from the past of variable Y. This is illustrated in Figure 2.3. Transfer entropy may be defined in terms of contitional entropy as shown in Equation 2.7:

$$TE_{X \to Y} = H(Y|Y_{past}) - H(Y|Y_{past}, X_{past}) \tag{2.7}$$

Transfer entropy may alternatively be defined in terms of conditional mutual information, per Equation 2.8. This is illustrated in Figure 2.4.

$$TE_{X \to Y} = I(X, X_{past}|Y_{past}) \tag{2.8}$$

The Y_{past} and X_{past} terms are usually approximated by taking the past value at some time $t - \delta t$ as a proxy for the entire past history:

$$\begin{aligned} \hat{Y}_{past} &= Y(t - \delta t) \\ \hat{X}_{past} &= X(t - \delta t) \end{aligned} \tag{2.9}$$

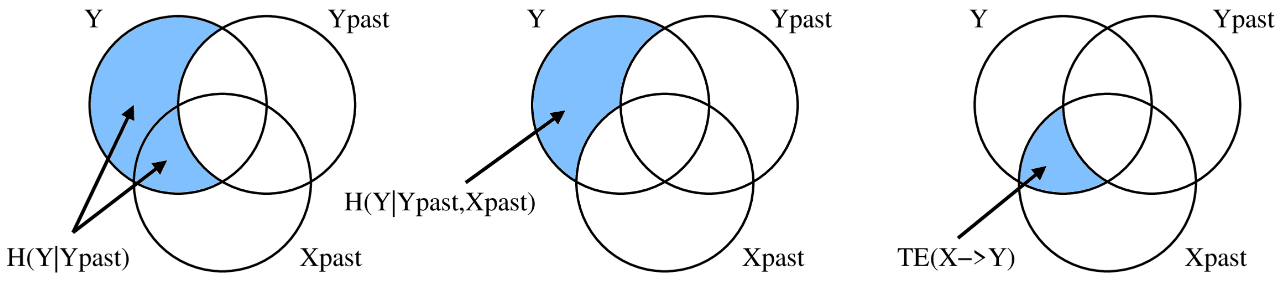


Figure 2.3: Transfer entropy between two signals, and its relation to conditional entropy.

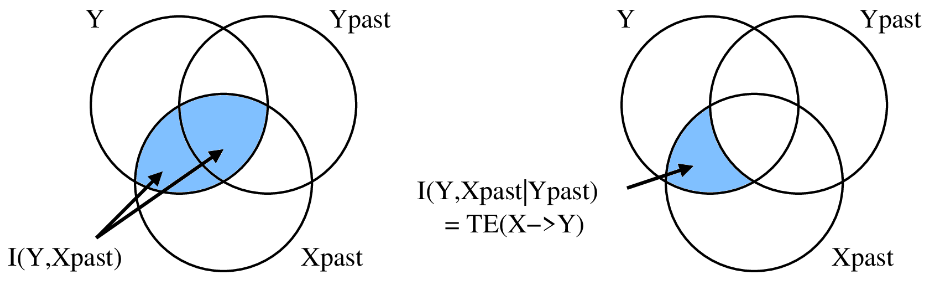


Figure 2.4: Transfer entropy between two signals, and its relation to mutual information.

Partial transfer entropy is used to disentangle the contributions of multiple variables to some Y variable. This is illustrated in Figure 2.5. Partial transfer entropy is defined by Equation 2.10:

$$pTE_{A\rightarrow Y} = H(Y|Y_{past}, B_{past}) - H(Y|Y_{past}, A_{past}, B_{past}) \tag{2.10}$$

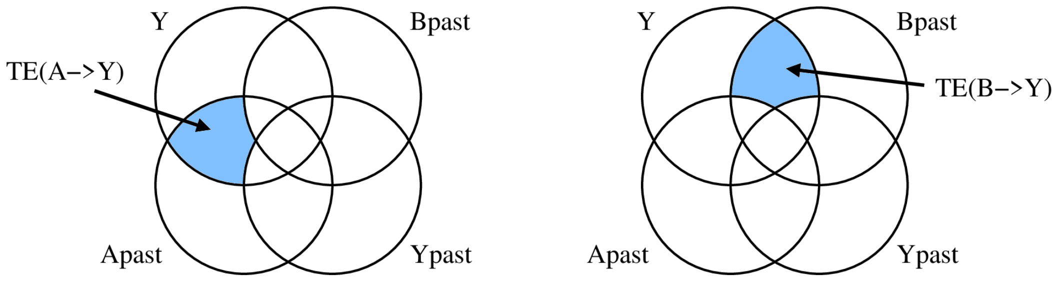


Figure 2.5: Partial transfer entropy between three signals.

2.5 References

- C. E. Shannon, *A Mathematical Theory of Communication*, The Bell System Technical Journal, v 27, pp 379–423,623-656, July, October, 1948

Chapter 3

Using This Library

Sample code for using this library is found in "`do_demo.m`" in the "`sample-code`" folder. An overview of how to perform the various operations is below.

- Data is expected to be supplied either as matrices with dimension $N_{chans} \times N_{samples}$, or as Field Trip data structures.
- To compute mutual information between two or more signals, use the `cEn_calcMutualInfo` family of functions.
- To compute time-lagged mutual information (mutual information between the present of a destination signal and the past of one or more source signals, use the `cEn_calcLaggedMutualInfo` family of functions.
- To compute transfer entropy from the past of one or more source signals to a destination signal, use the `cEn_calcTransferEntropy` family of functions. If two or more source signals are supplied, this gives the partial transfer entropy from each of them.
- The function versions with a "`FT`" suffix accept Field Trip data structures. The function versions with a "`MT`" suffix use a multithreaded implementation (faster but requires the Parallel Computing Toolbox).
- Some functions accept an "extrapolation parameter structure" as an optional argument. If this is *not supplied*, no extrapolation is performed. If this is "`struct()`", extrapolation is performed and default parameters are used.
- There are several different ways of specifying histogram bins for entropy calculations. These are discussed below.

A complete list of functions and a description of the extrapolation parameter structure can be found in the Library Reference document. A description of the extrapolation algorithm is given in Section 4.

For neuroscience data, the following considerations apply:

- For continuous-valued data like local field potentials, you'll need to decide on the number of bins used when building histograms (these are used as probability density functions). Mutual information requires building two-dimensional histograms (for two sources), transfer entropy requires three-dimensional histograms, and partial transfer entropy with two sources requires four-dimensional histograms. You need enough samples in the histogram bins for decent statistics, so the number of bins is usually kept very small (8 works well in the test code and sample code; literature typically uses even fewer than this).
- If you specify the number of bins to be used, the calculation functions choose bin edges for each signal being processed such that each bin ends up with the same number of samples in it. This is the usual approach to constructing histograms for mutual information and transfer entropy analysis. You can specify one bin count to be used for all signals, or specify one bin count per signal so that each can have a different number of bins.
- You can alternatively pass in a set of bin edges (one set per signal) as a cell array. This is mostly used when you want to perform several analyses and want to keep binning consistent between them.
- For discrete-valued data like spike counts, you can use `cEn_getMultivariateHistBinsDiscrete` to get a set of histogram bin edges that provides one bin per discrete value seen, for each input signal.
- The more bins and histogram dimensions you have, the more samples are needed to compute information statistics. The practical effect is that you're going to need to aggregate across a lot of trials to get useful statistics for anything beyond two-signal mutual information. Extrapolation reduces the number of samples needed by about $3\times$ to $10\times$.
- To measure uncertainty in information and entropy statistics, make a large number of surrogates via bootstrapping. This will let you determine if there are significant differences between trials from different experiment conditions. To measure significant changes with respect to a null condition, build a large number of surrogates via shuffling to get null case statistics. Anything involving "a large number of surrogates" will probably be the most time-consuming step of your analysis.

Chapter 4

Quadratic Extrapolation

This library's calculations of conditional entropy, mutual information, and transfer entropy are estimates. These estimates converge on the true values as the number of samples is increased. A rigorous analysis of the effect of small sample sizes on estimates typically performed with neural data is presented in Treves 1995. The take-away from that work is that entropy tends to be over-estimated, and that the estimate tends to converge monotonically on the correct value as sample size is increased. An example of this convergence for mutual information estimated using this library is shown in Figure 4.1 (left pane).

A follow-up analysis presented in Strong 1998 showed that for sample counts above a threshold, the true entropy value could be estimated by curve-fitting estimates from finite sample counts and extrapolating the curve fit to infinite sample counts. An example of applying this extrapolation method to mutual information estimation using this library is shown in Figure 4.1 (right pane). Note that the estimate converges more quickly but is no longer monotonic.

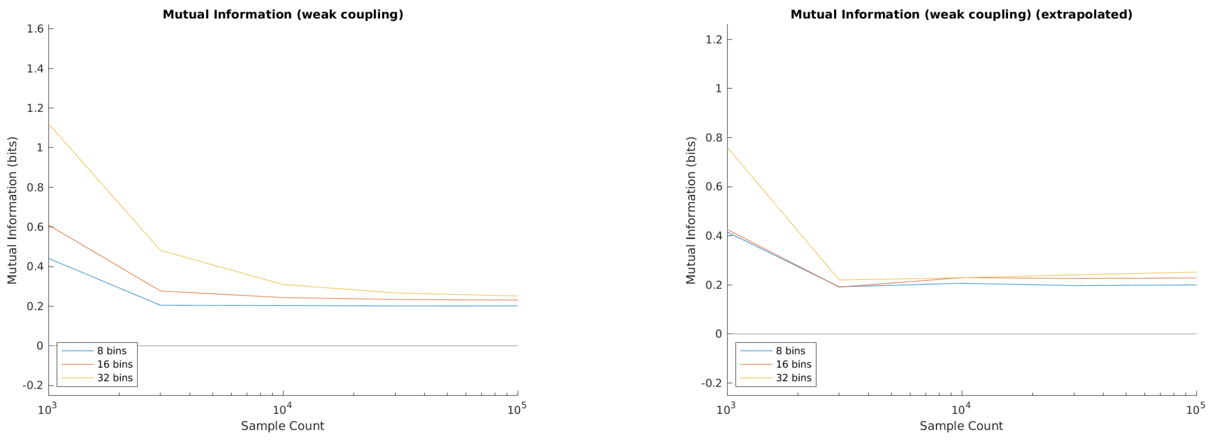


Figure 4.1: Mutual information between weakly coupled signals computed without extrapolation (left) and with extrapolation (right), as a function of the number of samples used to produce the estimate.

This library's implementation of the extrapolation algorithm is based on description given in Palmigiano 2017, used in that work for the estimation of transfer entropy. In this library it may be used (at the user's discretion) for the estimation of conditional entropy, mutual information, and transfer entropy.

The algorithm is supplied with several divisor values N_{div}. For each divisor value several random subsets of the data of size $N_{samp} \div N_{div}$ are chosen. The metric to be estimated is evaluated for each of these random subsets, and the results are averaged to produce an estimate associated with that N_{div}. A quadratic curve fit is performed as a function of N_{div}. This is used to estimate the metric value at $N_{div} = 0$, which corresponds to an infinite number of samples.

As noted in Strong 1998, there is a minimum sample count below which this estimation method fails. With or without quadratic extrapolation, it will be necessary to test several sample counts to evaluate whether the estimate has converged or not. In testing, the practical impact of extrapolation was to reduce the number of samples needed for convergence by approximately $3\times$ to $10\times$.

4.1 References

- A. Treves, S. Panzeri, *The Upward Bias in Measures of Information Derived from Limited Data Samples*, Neural computation, v 7, pp 399-407, 1995
- S. P. Strong, R. Koberle, R. R. de Ruyter van Steveninck, W. Bialek, *Entropy and Information in Neural Spike Trains*, Physical Review Letters, v 80, no. 1, pp 197-200, January 1998
- A. Palmigiano, T. Geisel, F. Wolf, D. Battaglia, *Flexible Information Routing by Transient Synchrony*, Nature Neuroscience, v 20, no. 7, pp 1014-1022, 2017