Convergent Cross Mapping (CCM) Notes

October 22, 2013

1 Basic Idea

Convergent cross mapping (CCM) is introduced in *Detecting Causality in Complex Ecosystems*[1] by Sugihara *et. al.*. The supplementary text for that paper is very helpful[2], although care should be taken when comparing the figure numbers between the text documents (the figure numbers in the supplementary text seem to be off by one).

CCM is a technique used to identify "causality" between time series and is intended to be useful in situations where the only other statistical "causality" measure, Granger causality, is known to be invalid (i.e. in dynamic systems that are "nonseperable"). The authors state that CCM is a "necessary condition for causation", but it may be best to avoid all discussion of causality in this work. It is well known (refs would be nice) that Granger causality is unrelated to causality as it is typically understood in physics. As such, rather than consider the necessity and sufficiency of CCM for causation, we will use the term "directed correlation".

CCM is very similar to simplex projection, which was introduced by Sugihara and May in *Nonlinear forecasting as a way of distinguishing chaos from measurement error*[3] and *Determining error from chaos in ecological time series*[4]. A concise overview of simplex projection can be found online[5]. Simplex projection uses the points with the most similar histories to the point at t to predict the point at $t+1$. Similarly, CCM uses points with the most similar histories to a point $X(t)$ (in a time series X) to estimate the point $Y(t)$ (in a time series Y).

2 Algorithm

The algorithm (as it has been implemented here in Matlab) consists of five steps:

1. Create the shadow manifold for X, called M_X
2. Find the nearest neighbours to $X(t)$ in M_X
3. Use the nearest neighbours to create weights
4. Use the weights to estimate $Y(t)$, called $Y(t)|M_X$
5. Find the correlation between $Y(t)$ and $Y(t)|M_X$ (this correlation is what is reported as the "directed correlation")

The steps vary in complexity and are explained in more detail below.

2.1 Create M_X

Given an embedding dimension E, the shadow manifold of X, called M_X, is created by associating an E-dimensional vector to each point $X(t)$ that is constructed as $\vec{X}(t) = (X(t), X(t-\tau), X(t-2\tau), \ldots, X(t-(E-1)\tau)$. The first such vector is created at $t = 1 + (E-1)\tau \equiv t_s$ and the last is at $t = L \equiv t_l$ where L is the time series length (or "library length"). The shadow manifold of X is the collection of all such vectors, i.e. $M_X = \{\vec{X}(t) | t \in [t_s, t_l]\}$. The time step τ is not discussed much in any of the Sugihara papers, and it appears to always be assumed that $\tau = 1$. We will follow that assumption throughout these notes unless specifically stated otherwise.

[1] http://www.uvm.edu/ cdanfort/csc-reading-group/sugihara-causality-science-2012.pdf

[2] http://www.sciencemag.org/content/suppl/2012/09/19/science.1227079.DC1/Sugihara.SM.pdf

[3] http://simplex.ucsd.edu/SugiMay-Chaos.pdf

[4] http://simplex.ucsd.edu/SugiGrenMay-Chaos.pdf

[5] http://simplex.ucsd.edu/

2.2 Find Nearest Neighbours

The minimum number of points required for a bounding simplex in an E-dimensional space is $E+1$ (find a non-Sugihara reference for this statement). Following this requirement, the $E+1$ nearest neighbours for a point $\vec{X}(t)$ on M_X (remember that this "point" on the shadow manifold is an E-dimensional vector of lagged time series points from X) are found. The distances d to these points and the times t at which they occur are recorded. Thus, the nearest neighbour search results in a set of distances $\{d_1, d_2, \ldots, d_{E+1}\}$ and an associated set of times $\{t_1, t_2, \ldots, t_{E+1}\}$ (where the subscript 1 denotes the closest neighbour, 2 denotes the next closest neighbour, etc.). The distances for a point $\vec{X}(t)$ are defined as

$$d_i = D\left(\vec{X}(t), \vec{X}(t_i)\right) \quad ,$$

where $D(\vec{a}, \vec{b})$ is the Euclidean distance between vectors $\vec{a}$ and $\vec{b}$ (implemented as `norm(a-b)` in the Matlab algorithm).

2.3 Create Weights

Each nearest neighbour will be used to find an associated weight. Define the unnormalized weights as

$$u_i = e^{-\frac{d_i}{d_1}} \quad .$$

In this way, each nearest neighbour will have a set of $E+1$ weights associated to the distance (and time) sets. The weights are defined as

$$w_i = \frac{u_i}{N} \quad ,$$

where the normalization factor is given as

$$N = \sum_j u_j \quad .$$

2.4 Find $Y(t)|M_X$

A point $Y(t)$ in Y can be estimated using the (normalized) distances to the points in X that have the most similar histories to the point $X(t)$ (i.e. using the weights calculated above). This estimate is calculated as

$$Y(t)|M_X = \sum_i w_i Y(t_i) \quad ,$$

where w_i are the weights calculated in the previous subsection and t_i are the times associated to the nearest neighbours (and, subsequently, the weights w_i).

2.5 Find the Directed Correlation

Define the directed correlation as

$$C_{YX} = \rho_{Y(t),Y(t)|M_X} \quad ,$$

where $\rho_{A,B}$ is the standard Pearson's correlation coefficient between A and B. It can be seen from the above algorithm that $X = Y \Rightarrow C_{YX} = C_{XY}$, but in general, $C_{YX} \neq C_{XY}$. Consider the following terminology:

- "Directionally correlated from X to Y" means $C_{YX} < C_{XY}$
- "Directionally correlated from Y to X" means $C_{YX} > C_{XY}$
- "Bidirectionally correlated from X to Y" means $C_{YX} = C_{XY}$
- "Uncorrelated from X to Y" means $C_{YX} = C_{XY} = 0$

The first two terms will be discussed in more detail in the following sections.

3 Point of Confusion