Causal Inference on Event Sequences

Kailash Budhathoki * Jilles Vreeken *

Abstract

Given two event sequences, i.e. two discrete valued time series, of length n can we tell whether they are causally related? That is, can we tell whether x^n causes y^n, whether y^n causes x^n? Can we do so without having to make assumptions on the distribution of these time series, or about the lag of the causal effect? And, importantly for practical application, can we do so accurately and efficiently? These are exactly the questions we answer in this paper.

We propose a causal inference framework for event sequences based on information theory. We build upon the well-known notion of Granger causality, and define causality in terms of compression. We infer that time series x^n is likely a cause of time series y^n if y^n can be compressed (much) better given the past of x^n than the other way around. To compress the event sequences, we use minimax optimal codes with respect to a parametric family of distributions. To show this works in practice, we propose CUTE, a linear time method for inferring the causal direction between two event sequences. Empirical evaluation shows that CUTE works well in practice, is much more robust than transfer entropy, and ably reconstructs the ground truth on river flow and spike train data.

1 Introduction

Telling cause from effect from data that was not collected through carefully controlled randomized trials, i.e. observational data, is one of the fundamental problems in science. We consider the case where we are given two discrete-valued time series—event sequences—of length n, and have to determine whether it is more likely that x^n caused y^n, or the other way around, that y^n caused x^n.

Perhaps the most well-known framework for causal inference on time series is Granger causality, which postulates that x^n is likely a cause of y^n if the past values of x^n help to significantly better predict the values of y^n than we can do with just the data over the past of y^n. Typically predictability is measured in terms of variance of the error. While this framework makes intuitive sense, to put it to practice it does require us to make assumptions on the generating process, as well as on the measure of predictability.

In this paper we take a related, but subtly different approach. We take an information theoretic viewpoint and define causality in terms of compression. Simply put, we say that X causes Y if we save more bits by compressing the data of Y with additionally the past of X, than vice versa. The larger the difference, the more confident we are. To optimally compress the data, we would need to know its distribution. In practice, however, we only have observed data and a class of possible prediction strategies—in which the true distribution may or may not live. We hence build our inference framework on the notion of sequential normalized maximum likelihood (SNML), which is a strategy that is guaranteed to give the minimum number of additional bits (regret) compared to the true distribution, regardless of input, and regardless of whether or not the true distribution is in the model class under consideration. At every time step, our prediction for the current outcome is proportional to the Maximum Likelihood estimate of the overall sequence, including the past outcomes as well as the current one.

We give the general theory for causal inference on event sequences using SNML, including a detailed exposition on how to derive our causal indicators for binary event sequences based on the class of binomial distributions. The extension to multinomial distributions is trivial. Importantly, for discrete data in general, CUTE, which stands for **c**a**u**sal inference on even**t** sequenc**e**s, has only a linear time worst case runtime complexity.

While there exist many causal inference approaches for time series, many of which based on Granger causality, there are only few that are applicable on event sequences. Transfer Entropy, which is based on Shannon Entropy, is perhaps the most well known. Conceptually, it is closely related, as lower entropy corresponds to better compression. Unlike CUTE, however, it only considers one model, not a complete model class, and has no minimax optimality guarantee. Moreover, unlike CUTE, but like many frameworks it requires the user to set a lag parameter that specifies the (expected) lag between the cause and the effect, and its computational complexity is exponential in this parameter. We refer the reader to Sec. 3 for a more in depth discussion of related work.

We empirically evaluate CUTE on a wide range of binary-valued event sequences. Results on synthetic data show that unless the effect is only time-shifted, it outperforms transfer entropy by a wide margin. Additionally, we consider two case studies on real world data, where we find that CUTE with high accuracy reconstructs the ground truth in water elevation levels in two rivers, as well as in discovering excitatory connections in neural spike train data. The code of CUTE is freely available for research purposes.

*Max Planck Institute for Informatics and Saarland University.
{kbudhath,jilles}@mpi-inf.mpg.de

2 Theory

In this section, we formally introduce the problem, and present our framework.

2.1 The Problem Let $x^n = x_1, x_2, \ldots, x_n$ be an event sequence, a time series of n observed outcomes where each outcome x_i is an element of a discrete space of observations $\mathcal{X} \in \{1, 2, \ldots, m\}$. Likewise $y^n = y_1, y_2, \ldots, y_n$ such that $y_i \in \mathcal{Y}$. Given two correlated event sequences x^n and y^n, we are interested in finding the most likely causal direction between them. That is, we would like to identify whether x^n causes y^n, or y^n causes x^n, or they are just correlated.

2.2 Assumptions To measure the causal dependence between two event sequences, we take the usual assumptions of Granger causality [4]. Namely, we assume the following.

Assumption 1. *Cause precedes the effect in time.*

Assumption 2. *Cause has unique information about the future values of effect.*

Assumption 1 is commonly accepted [2, 15], and also corroborated by the thermodynamic principle—the arrow of causation points in the same direction as the arrow of time. That is, the past influences the future, but not the other way around. One of the implications of Assumption 1 is that we assume there is no confounding event sequence z^n that is the common cause of both x^n and y^n. The other implied assumption is that there is no instantaneous causal relationship—the present value of the cause does not help in the prediction of the present value of the effect.

Assumption 2 is also intuitively plausible: the past of the cause and the future of the effect should share some information which cannot be accounted for only by the knowledge of the past of the effect. This also means that causal dependence measure should be able to quantify that unique information which is not available otherwise.

2.3 Measuring Causal Dependence We base our causal dependence measure on the foundation of Granger causality [4] where causal dependence is measured in terms of *predictability.*

Definition 1. (Granger Causality) *Let $\mathcal{I}^t$ be the information available as of time t in the entire universe that includes both x^{t-1} and y^{t-1}, and $\mathcal{I}^t_{\neg x}$ be that in a modified universe where x^{t-1} is excluded. We say that x^t Granger-causes y^t if*

$$P(y_{t+1} \mid \mathcal{I}^t) > P(y_{t+1} \mid \mathcal{I}^t_{\neg x}) \ ,$$

where P indicates the prediction.

In the original paper [4], predictability is measured in terms of the variance of the error in regression, thereby ending up with a reverse inequality.

We associate predictability with compression. In particular, we consider the encoded length of the event sequence using a prediction strategy. Intuitively the more predictable an event sequence is, the smaller the number of bits required to describe it using the prediction strategy.

Let $P(x_t \mid x^{t-1})$ be the prediction of current outcome x_t given its past x^{t-1}. To encode the entire event sequence x^n, we use the prediction $P(\cdot \mid x^{t-1})$ at every iteration $t = 1, 2, \ldots, n$. Let $P(\mathcal{X}^n)$ be the probability distribution over all the possible event sequence of size n from the domain $\mathcal{X}$, and $P(\mathcal{X}^n = x^n)$ be the probability mass function for event sequence x^n. Then the predictions $P(\cdot \mid x^{t-1})$ can be considered as a conditional of the joint distribution, i.e. $P(\mathcal{X}^n = x^n) = \prod_{t=1}^{n} P(x_t \mid x^{t-1})$.

The ideal code length for encoding the current outcome x_t given its past x^{t-1} using the prediction $P(x_t \mid x^{t-1})$ is $-\log P(x_t \mid x^{t-1})$. In learning theory, it is commonly known as *log loss.* Hence the total encoded length of the event sequence x^n using its past, denoted $\ell(x^n)$, is given by

$$\ell(x^n) = \sum_{t=1}^{n} -\log P(x_t \mid x^{t-1}) \ .$$

Likewise, let $P(x_t \mid x^{t-1}, y^{t-1})$ be the prediction probability of x_t given the past outcomes of x^n, as well as the past outcomes of y^n. The total encoded length of the event sequence x^n using its past as well as the past of y^n, denoted $\ell(x^n \mid y^n)$, is then

$$\ell(x^n \mid y^n) = \sum_{t=1}^{n} -\log P(x_t \mid x^{t-1}, y^{t-1}) \ .$$

Note that the encoded size $\ell(x^n)$ measures the predictability of x^n from its past outcomes, and $\ell(x^n \mid y^n)$ measures the predictability of x^n from its past, as well as the past of y^n. Their difference, hence, measures the extra predictability of x^n contributed by the past of y^n which is not available otherwise. With that, we define the causal dependence from the direction y^n to x^n as

$$\Delta_{y^n \to x^n} = \ell(x^n) - \ell(x^n \mid y^n) \ ,$$

and that from x^n to y^n is given by

$$\Delta_{x^n \to y^n} = \ell(y^n) - \ell(y^n \mid x^n) \ .$$

Due to the dependence in *time*, our causal dependence measure is inherently asymmetric. Under our assumptions, the direction with larger dependence is likely the true causal direction. Thus, using the above indicators we arrive at the following causal inference rules on event sequence data.

- If $\Delta_{x^n \to y^n} > \Delta_{y^n \to x^n}$, we infer $x^n \to y^n$.
- If $\Delta_{x^n \to y^n} < \Delta_{y^n \to x^n}$, we infer $y^n \to x^n$.

- If $\Delta_{x^n \to y^n} = \Delta_{y^n \to x^n}$, we are undecided.

That is, if the added knowledge of the past outcomes of x^n makes the encoding of y^n easier than vice versa, we infer x^n is likely the cause of y^n. If it is the other way around, we infer y^n is likely the cause of x^n. If causal dependence is the same in both directions, we remain undecided.

The larger the difference in causal dependence in both directions, the more confident we are. In practice, we can always introduce a threshold τ on the absolute difference between two indicators $|\Delta_{x^n \to y^n} - \Delta_{y^n \to x^n}|$, and treat the results smaller than τ as undecided.

The proposed causal inference rule, however, is based on the premise that we have access to the *true* distribution P. In practice, we of course do not know this distribution, we only have *observed* data, and possible models or prediction strategies $\mathcal{P}$. The true distribution *may* or *may not* be in the models under consideration. Next we discuss how to construct a prediction strategy from the class such that we get optimal performance, regardless of whether true distribution lies in $\mathcal{P}$.

2.4 Sequential Normalised Maximum Likelihood As models, prediction strategies $\mathcal{P}$, we consider parameterised families of distributions. Formally, we define $\mathcal{P}$ as

$$\mathcal{P} = \{P_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\} ,$$

where Θ is a parameter space, i.e. $\Theta = \{\boldsymbol{\theta} \in \mathbb{R}^k\}$, and $k > 0$.

Typically the performance of a prediction strategy P on an event sequence x^n w.r.t. a model class $\mathcal{P}$ is measured by *regret*, which is defined as

$$\begin{aligned}\mathcal{R}(P; x^n) &= \sum_{t=1}^{n} -\log P(x_t \mid x^{t-1}) - \min_{P_\theta \in \mathcal{P}} \sum_{t=1}^{n} -\log P_{\boldsymbol{\theta}}(x_t \mid x^{t-1}) \\ &= -\log P(x^n) - \min_{P_\theta \in \mathcal{P}} \left(-\log P_{\boldsymbol{\theta}}(x^n)\right) \quad .\end{aligned}$$

In words, regret is the additional number of bits required to encode the event sequence using a prediction strategy P instead of the best prediction strategy from the model class $\mathcal{P}$. The regret, however, is not the same for all $x^n \in \mathcal{X}^n$—it can be small for some, and large for others. As we want to be robust with regard to model misspecification, we therefore consider the worst-case regret over all possible event sequences of length n,

$$\mathcal{R}_{\max}(P; n) = \max_{x^n \in \mathcal{X}^n} \mathcal{R}(P; x^n) .$$

The optimal prediction strategy relative to a model class $\mathcal{P}$ for a sample of size n is then the one that minimises the worst-case regret,

$$\min_P \mathcal{R}_{\max}(P; n) .$$

If the true data generating distribution lies in the model class under consideration $\mathcal{P}$, the maximum likelihood (ML) strategy—predict the next outcome x_{t+1} using the distribution $P_{\hat{\boldsymbol{\theta}}(x^t)}$ with $\hat{\boldsymbol{\theta}}(x^t)$ being the ML estimator based on the past outcomes x^t—will be the optimal prediction strategy. The ML strategy, however, is not robust against the misspecification of the model class, i.e. when the true distribution is not in the model class under consideration the result can be arbitrarily bad [7].

We would like to have a prediction strategy P that is optimal regardless of whether the true distribution lies in $\mathcal{P}$. Surprisingly a slight modification of the ML strategy can achieve the optimality, and gives the solution to the minimax problem posed above. The modification involves computing the ML estimator of the data sequence including the current outcome, followed by the normalisation of the distribution. That is, the modified strategy predicts x_t with a distribution proportional to $P_{\hat{\boldsymbol{\theta}}(x^{t-1}, x_t)}$, where $\hat{\boldsymbol{\theta}}(x^{t-1}, x_t)$ is the ML estimator for the data sequence $x_1, \ldots, x_{t-1}, x_t$, and defined as

$$P_{\text{SNML}}(x_t \mid x^{t-1}) = \frac{P_{\hat{\boldsymbol{\theta}}(x^{t-1}, x_t)}}{\sum_{x \in \mathcal{X}} P_{\hat{\boldsymbol{\theta}}(x^{t-1}, x)}} .$$

This strategy is also known as the Sequential Normalised Maximum Likelihood model (SNML) [7, 11]. We use it to encode the event sequence. For exponential family of distributions (e.g. Bernoulli, Multinomial, Gaussian, etc.), we can use their respective closed-form expression to calculate the ML estimator $\hat{\boldsymbol{\theta}}$. Hence, it is easy to compute the SNML strategy for the whole exponential family.

Importantly the SNML strategy is general in the sense that we are only restricted by the choice of our model class. For clarity, we focus specifically on binary data. Without loss of generality, it generalises to the general discrete case.

2.5 SNML for Binary Data As models for binary data, we consider a parameterised family of Bernoulli distributions. The parameterised family of Bernoulli distributions $\mathcal{B}$ is defined as $\mathcal{B} = \{P_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\}$, where Θ is a parameter space defined as $\Theta = \{\boldsymbol{\theta} \in [0, 1]\}$. The probability mass function for Bernoulli distribution is given by

$$P_{\boldsymbol{\theta}}(X = k) = \boldsymbol{\theta}^k (1 - \boldsymbol{\theta})^{1-k} ,$$

where $k \in \{0, 1\}$. The ML estimator for an event sequence x^{t-1} relative to the Bernoulli class is given by $\hat{\boldsymbol{\theta}}(x^{t-1}) = t_1/(t-1)$, where $t_1 = \sum_{i=1}^{t-1} x_i$ is the number of ones in x^{t-1}. Let $t_0 = t - 1 - t_1$ be the number of zeros in x^{t-1}. Then the denominator of the SNML strategy for predicting x_t given

the past x^{t-1} is given by

$$\begin{aligned}
&\sum_{x\in\mathcal{X}} P_{\hat{\theta}(x^{t-1},x)} = \sum_{x\in\{0,1\}} P_{\hat{\theta}(x^{t-1},x)} = P_{\hat{\theta}(x^{t-1},0)} + P_{\hat{\theta}(x^{t-1},1)} \\
&= (\hat{\theta}(x^{t-1},0))^{t_1}(1-\hat{\theta}(x^t,0))^{t_0+1} + \\
&\quad (\hat{\theta}(x^{t-1},1))^{t_1+1}(1-\hat{\theta}(x^t,1))^{t_0} \\
&= \left(\frac{t_1}{t}\right)^{t_1}\left(1-\frac{t_1}{t}\right)^{t_0+1} + \left(\frac{t_1+1}{t}\right)^{t_1+1}\left(1-\frac{t_1+1}{t}\right)^{t_0} \\
&= \frac{1}{t^t}\left\{t_1^{t_1}(t_0+1)^{t_0+1} + (t_1+1)^{t_1+1}t_0^{t_0}\right\} \ .
\end{aligned}$$

Thus the prediction for the outcome $x_t = 1$ from its past x^{t-1} using the SNML strategy is given by

$$\begin{aligned}
P_{\text{SNML}}(x_t = 1 \mid x^{t-1}) &= \frac{P_{\hat{\theta}(x^{t-1},1)}}{\sum_{x\in\mathcal{X}} P_{\hat{\theta}(x^{t-1},x)}} \\
&= \frac{(t_1+1)^{t_1+1}t_0^{t_0}}{t_1^{t_1}(t_0+1)^{t_0+1} + (t_1+1)^{t_1+1}t_0^{t_0}} \ , \qquad (2.1)
\end{aligned}$$

and that for $x_t = 0$ is trivially given by

$$P_{\text{SNML}}(x_t = 0 \mid x^{t-1}) = 1 - P_{\text{SNML}}(x_t = 1 \mid x^{t-1}) \ .$$

In practice, instead of computing the SNML prediction, which could possibly result in overflow errors for large sample size, we can directly compute the SNML code length using the *log-sum-exp* trick. For our purpose, we also need to compute the encoded length of the event sequence x^n given the other event sequence y^n. Next we discuss how to conditionally encode one event sequence given the other using their past.

2.5.1 Conditional Compression To encode an event sequence x^n given a times series y^n, i.e. for $\ell(x^n \mid y^n)$, we have to compute $-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$. To predict the outcome x_t, we can use either x_i or y_i in every time step $i = 1, 2, \ldots, t-1$. Let $u = \sum_{i=1}^{t-1} x_i \oplus y_i$, with $\oplus$ being the Boolean XOR operator, be the number of time steps where the outcome of x_i and y_i differ. Thus we end up with 2^u different event sequence for predicting the outcome x_t. Among all possibilities, we choose the one that improves the prediction of x_t.

For exposition, we present a toy example in Eq. (2.2). Suppose we want to predict the outcome x_4 given its past $x^3 = 111$, and that of y^n, which is $y^3 = 010$. At every time step—except for the second— we have two choices. Overall we therefore can construct four different event sequence $z_1, \ldots, z_4$ that we can use to base our prediction on.

$$\begin{array}{lcccccc}
x^3 & : & 1 & 1 & 1 & \\
y^3 & : & 0 & 1 & 0 & \\
z_1 & : & 0 & 1 & 0 & (y_3, 1, y_1) \\
z_2 & : & 0 & 1 & 1 & (y_3, 1, x_1) \\
z_2 & : & 1 & 1 & 1 & (x_3, 1, x_1) \\
z_4 & : & 1 & 1 & 0 & (x_3, 1, y_1)
\end{array} \qquad (2.2)$$

By virtue of Eq. (2.1), we know that the prediction depends on the number of ones in the past t_1. Therefore we can use the number of ones present in newly constructed event sequence z_is to directly get the prediction. Let $t_x = \sum_{i=1}^{t-1} x_i$ be the number of ones in x^{t-1}, defining t_y analogous. To predict x_t given its past and that of y_n, we can use the number of ones in the range from $t_{\min} = \min(t_x, t_y)$ to $t_{\max} = \sum_{i=1}^{t-1} x_i \vee y_i$, where $\vee$ is the boolean OR operator.

In every iteration, we choose the number of ones $t_1 \in \{t_{\min}, t_{\min}+1, \ldots, t_{\max}\}$ that results in the minimal code length $-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$, using the prediction $P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$. This way, the optimisation of $-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$ also includes the index set for that of the $-\log P_{\text{SNML}}(x_t \mid x^{t-1})$. Thus we have

$$-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1}) \leq -\log P_{\text{SNML}}(x_t \mid x^{t-1}) \ .$$

The equality holds when y^{t-1} does not help in the prediction of x_t. As a result, both causal indicators are positive, i.e. $\Delta_{x^n \to y^n} \geq 0$, and $\Delta_{y^n \to x^n} \geq 0$.

At first glance, it is not evident whether the prediction is monotone with respect to the number of ones in the past t_1. Moreover derivative analysis, or inductive proof appears to be non-trivial. Therefore we numerically compute the code length of the outcome x_t using the prediction $P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$ for various number of ones $t_1 = 1, 2, \ldots, t-1$ for a fixed t. Further we repeat the same process for $t = 1, 2, \ldots, 50$. In Fig. 1, we show the results in a 3D plot. We observe that the code length for the outcome $x_t = 1$ is monotonically decreasing with respect to the number of ones in the past t_1 for a fixed t. The code length for the outcome $x_t = 0$, however, is monotonically increasing relative to t_1 for a fixed t.

This numerical analysis suggests that $t_{\max}$ maximises the prediction of the outcome $x_t = 1$ given its past x^{t-1}, and that of y^n. On the contrary, $t_{\min}$ maximises the prediction of the outcome $x_t = 0$. Hence, the prediction for the outcome $x_t = 1$ from its past x^{t-1}, and that of y^n using SNML strategy is given by

$$P_{\text{SNML}}(x_t = 1 \mid x^{t-1}, y^{t-1}) = \frac{Z}{t_{\max}^{t_{\max}}(t_0+1)^{t_0+1} + Z} \ , \qquad (2.3)$$

where $Z = (t_{\max}+1)^{t_{\max}+1}t_0^{t_0}$, and $t_0 = t - 1 - t_{\max}$. Likewise, the prediction for the outcome $x_t = 0$ from its past x^{t-1}, and

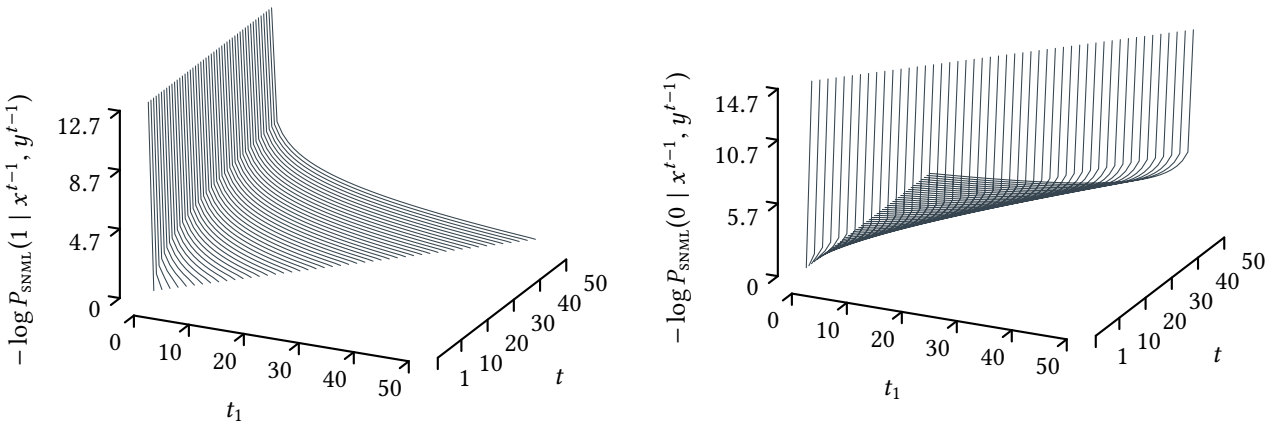

Figure 1: SNML predictions using the past of x^n, and that of y^n. For time steps $t = 1, 2, \ldots, 50$, and number of ones in the past $t_1 = 1, 2, \ldots, 50$ such that $t_1 < t$, we plot the code length using the SNML prediction strategy for (left) $x_t = 1$, and (right) $x_t = 0$.

that of y^n using SNML strategy is given by

$$P_{\text{SNML}}(x_t = 0 \mid x^{t-1}, y^{t-1}) = \frac{K}{K + t_{\min}^{t_{\min}}(t_0 + 1)^{t_0+1}} , \tag{2.4}$$

where $K = t_{\min}^{t_{\min}}(t_0 + 1)^{t_0+1}$, and $t_0 = t - 1 - t_{\min}$.

From here onwards, we refer to our proposed framework as CUTE, for **ca**u**s**al inference on even**t** sequ**e**nces using SNML. All logarithms are to base 2, and by convention we use $0 \log 0 = 0$.

2.6 Computational Complexity To compute $\ell(x^n)$, we have to compute $-\log P_{\text{SNML}}(x_t \mid x^{t-1})$ for $t = 1, 2, \ldots, n$. In every iteration, we can keep track of the count of the number of ones t_1 we have seen so far. Given t and t_1, we can compute $-\log P_{\text{SNML}}(x_t \mid x^{t-1})$ in constant time, $O(1)$, using the closed form expression in Eq. (2.1). Therefore we can compute $\ell(x^n)$ in $O(n)$ time.

To compute $\ell(x^n \mid y^n)$, we have to compute $-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$ for $t = 1, 2, \ldots, n$. In every iteration, we can keep track of the count of number of ones t_x, t_y, and $t_{\max}$. Given t_x, t_y, and $t_{\max}$, we can compute $-\log P_{\text{SNML}}(x_t \mid x^{t-1}, y^{t-1})$ in constant time, $O(1)$, using the closed form expression in Eq. (2.3), and Eq. (2.4). Hence we can compute $\ell(x^n \mid y^n)$ in $O(n)$ time. This implies we can compute $\Delta_{y^n \to x^n}$ in $O(n)$ time.

Altogether the worst case computational complexity of the framework is $O(n)$.

3 Related Work

Causal inference techniques on time series, for the most part, are based on Granger causality [4]. The key idea is that a time series x^n does not Granger cause a time series y^n if the past of x^n does not help in predicting y^n given the past of y^n. Typically predictability in measured in terms of the variance of the error in regression. This also corresponds to a significance test assuming a multivariate time series model [3, 10]. There exists many variants of Granger causality depending on the assumed model, and the predictability measure.

Linear Granger causality, for instance, considers a vector autoregressive (VAR) model. A VAR model describes the current outcome as a linear function of its past values, and an additive error term. Non-linear Granger causality is an extension of Granger causality to non-linear systems [2]. The key idea there is to apply linear regression for each local neighbourhood and average the resulting statistical quantity over the entire attractor (a set of numerical values toward which a system tends to evolve). Wax & Rissanen [12] proposed a compression-based framework for measuring mutual and causal dependence on the foundations of Granger causality, with an instantiation for continuous real-valued data. Another variant of Granger causality is the transfer entropy (shortly TENT) [14] which measures predictability in terms of Shannon entropy. Transfer entropy can, unlike others, detect both linear and non-linear causal influences.

There do exist techniques that take a different approach from Granger causality. Chu & Glymour [3] propose conditional independence test on non-iid setting, and introduce the additive non-linear time series models (ANLTSM). It uses additive model regression as a general purpose non-linear conditional independence test. TS-LiNGAM [6] considers the general case where causal influences can occur either instantaneously or with considerable time lags. It combines the non-Gaussian instantaneous model with autoregressive models for causal analysis. Peters et al. [8] use restricted structural equation models, ANMs in particular, to find causal structures from time series. Huang & Kleinberg [5] introduce a causal inference framework based on

logical formulas where cause is a discrete variable, and effect is a continuous-valued variable.

All the frameworks above except transfer entropy, however, either work with continuous-valued or mixed time series data. The known variants of Granger causality for event sequences, or discrete-valued time series are either highly tailored for the problem at hand [10], or just minor variations [9]. Transfer entropy is close in spirit to CUTE, but unlike transfer entropy, we are robust to adversarial settings (e.g. misspecified model classes). Transfer entropy requires a lag parameter, whereas CUTE does not require one. Importantly CUTE runs in $O(n)$ time. TENT, on the other hand, takes $O(nk + 2^k)$, where k is the lag value. As it is the most commonly applied Granger causality framework, we compare CUTE against TENT in the experiments.

4 Experiments

We implemented CUTE in Python and provide the source code for research purposes, along with the used datasets, and synthetic dataset generator.[1] All experiments were executed single-threaded on Intel Xeon E5-2643 v3 machine with 256 GB memory running Linux. We compare CUTE with transfer entropy, shortly TENT, which is commonly regarded as the state of the art in Granger causality, and easily applicable to event sequences.

4.1 Synthetic Data To evaluate CUTE on data with the known ground truth, we use synthetic data. We generate the first 10 time steps of cause time series with a coin flip. In the successive time steps, we randomly choose two time points from the past 10 time steps, and apply the XOR operation on the outcomes that are between the selected time points. Unless stated otherwise, we sample 5000 points for both cause and effect.

Effect as a time-shifted version of cause We generate the effect time series by shifting the cause time series by k time steps forward, where k is randomly chosen between 1 and 10. At every time step of the resulting time series, we *flip* the outcome depending on the noise level. As for the first $k - 1$ time steps, we use coin flip to generate the outcomes. At each noise level, we generate 200 cause-effect pairs.

Unlike CUTE, TENT requires a *lag* parameter. Therefore we take the best result (maximum transfer entropy) obtained by varying the lag between 1 and 20 on every cause-effect pair. In Figure 2, we compare the accuracy of CUTE against transfer entropy (TENT) at various noise levels. We observe that CUTE performs at least as good as TENT at every level of noise.

[1] https://goo.gl/JErNG2

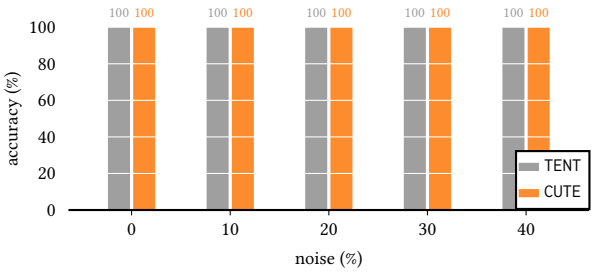

Figure 2: Accuracy vs. noise on synthetic cause-effect pairs, where effect is a *timeshifted* version of the cause.

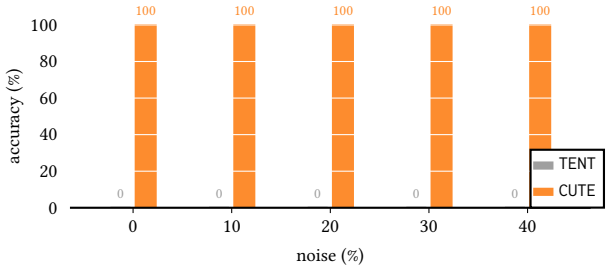

Figure 3: Accuracy vs. noise on synthetic cause-effect pairs, where effect is a *timeshifted*, and *inverted* version of the cause. TENT is undecided in all cases.

Effect as a time-shifted and inverted version of cause In the next experiment, we generate the effect time series by first shifting the cause time series by k time steps forward, and then *inverting* the resulting time series. The value of k is chosen randomly from the range 1 to 10. To assess the performance of the inference frameworks, we repeat the same process as in the previous experiment.

In Figure 3, we compare the accuracy of CUTE against transfer entropy (TENT) at various noise levels. We observe that CUTE is very accurate, and identifies the true direction at every level of noise, whereas TENT remains indecisive in all cases.

Effect as an XOR of its past as well as that of the cause Next to generate the effect time series, at every time step we randomly choose two time points from the past k time steps of the cause, and past l time steps of the effect, and apply the XOR operation on the outcomes of the past of both cause and effect that are between the selected time points. The value of k, and l are chosen randomly from the range 1 to 10. To assess the performance of the inference frameworks, we repeat the same process as in the previous experiment.

In Figure 4, we compare the accuracy of CUTE against transfer entropy (TENT) at various noise levels. We observe that CUTE outperforms TENT in all cases.

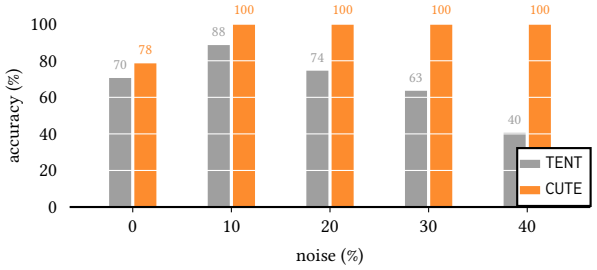

Figure 4: Accuracy vs. noise on synthetic cause-effect pairs, where effect is an *XOR* of its past, and that of the cause.

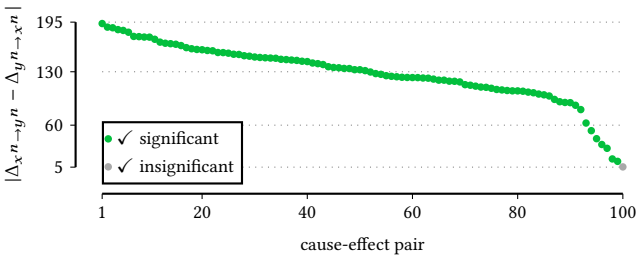

Figure 5: Synthetic cause-effect pairs, where effect is a shifted version of the cause with 10% noise, sorted by their corresponding difference in compression in two directions. We apply Benjamini-Hochberg correction to control the false discovery rate at a significance level of $\alpha = 0.01$.

Statistical Significance The problem of differentiating two directions (between $x^n \to y^n$ and $y^n \to x^n$) can be cast as an *identity testing* problem, which is one of the fundamental problems in statistics. As our method is based on compression, we can use the compression based identity testing framework [13] to assess the significance of inferred results. The framework can be roughly described as follows:

Definition 2. (Compression-based Identity Testing [13]) *Let x^n be a sequence over an alphabet Σ. Let H_0 be the null hypothesis that the source of x^n has a distribution P, and H_1 be the alternative hypothesis that the source of x^n has a distribution Q. We reject the null hypothesis H_0 if*

$$-\log P(x^n) - \{-\log Q(x^n)\} > -\log \alpha \ ,$$

where α is the level of significance. For each distribution P, and Q, the Type I error of the compression-based identity testing is not larger than α.

On a cause-effect pair (x^n, y^n) where CUTE makes a decision (say $x^n \to y^n$), we want to assess whether the decision is significant. To this end, our null hypothesis H_0 will be the joint distribution under the alternative direction $(y^n \to x^n)$. Then the alternative hypothesis H_1 will be that under the inferred direction. In that case, we reject H_0 if $\Delta_{y^n \to x^n} - \Delta_{x^n \to y^n} > -\log \alpha$.

In order to control the false discovery rate for multiple hypothesis testing, we use Benjamini-Hochberg procedure [1]. Let $H_1, H_2, \ldots, H_m$ be the null hypotheses tested, and $p_1, p_2, \ldots, p_m$ their corresponding p-values. We sort the p-values in ascending order. For significance level of α, we find the largest k such that $p_k \leq \frac{k}{m}\alpha$. We reject the null hypothesis for all H_i, where $i = 1, \ldots, k$.

We sample 100 cause-effect pairs, where effect is a shifted version of the cause with 10% noise. For each pair, we sample 1000 outcomes. In Figure 5, we sort the cause-effect pairs by their corresponding difference in causal dependence in two directions $|\Delta_{x^n \to y^n} - \Delta_{y^n \to x^n}|$. That also corresponds to sorting the pairs by their p-values in ascending manner. At a significance threshold of $\alpha = 0.01$ after applying Benjamini-Hochberg correction, we observe that all the inferences are correct, amongst which only one inference is insignificant.

4.2 Real Data

Next we investigate CUTE on real data.

River Water Level First we look into water level of rivers in Germany. We consider two rivers in particular: *Saar* and *Rhein*. For a river, we collect the raw water level recorded every 15 minutes in past one month from various water level recording stations.[2] This way we end up with 2880 data points from one station. The raw water level, however, is continuous real-valued, and hence we have to binarise the data. To this end, if the water level goes up from previous recording, we use 1. Otherwise we use 0. It is intuitively plausible to consider that the water level recording of the station upstream causes the water level recording of the station downstream.

For the *Saar* river, we collect the raw water level data from three stations, namely *Fremersdorf*, *Hanweiler*, and *Sankt Arnual*. Then we binarise the recordings from each station. In Figure 6, we show the map of the *Saar* river along with the recording stations. For instance, *Hanweiler* station is upstream compared to *Fremersdorf* station. Therefore the ground truth would be *Hanweiler* causes *Fremersdorf*. We run CUTE on all the pairs. In Figure 7, we present the results as a directed acyclic graph (DAG). The results clearly corroborate our intuition.

For the *Rhein* river, we collect the raw water level data from four stations, namely *Speyer*, *Mannheim*, *Worms*, and *Mainz*. Then we binarise the recordings from each station. In Figure 6, we show the map of the *Rhein* river along with the recording stations. After running CUTE on all the pairs, we end up with a DAG as shown in Figure 8. We see that

[2] http://www.pegelonline.wsv.de/webservices/files/Wasserstand+Rohdaten/

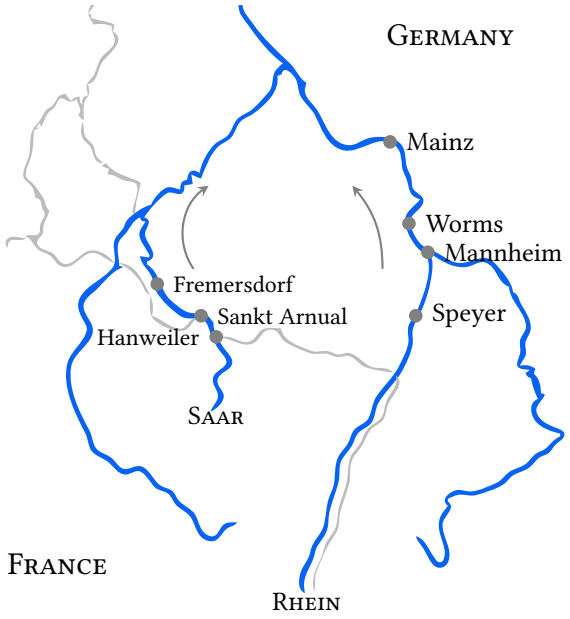

Figure 6: Map of the *Saar* and the *Rhein* river in Germany. The recording stations are marked with gray dots.

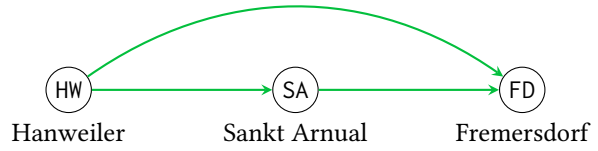

Figure 7: Results of CUTE on the *Saar* river. A green edge represents a correct causal direction.

CUTE identifies the correct direction in all but one case.

Temperature The *Temperature* dataset is the 48$^{\text{th}}$ pair in the Tübingen cause-effect benchmark pairs.[3] It contains indoor (x^n), and outdoor (y^n) temperature measurements recorded every 5 minutes. There are $n = 168$ measurements. We binarise the data like before. The ground truth of the pair is $y^n \to x^n$. With CUTE, we get the following scores:

$$\Delta_{x^n \to y^n} = 28.36, \ \Delta_{y^n \to x^n} = 41.81 \ .$$

Hence CUTE infers $y^n \to x^n$, which is in agreement with the ground truth.

Overall, these results show that CUTE finds sensible causal directions from real data.

Neural Spike Train Recordings Next we look at the neural spike train recordings data from an experiment

[3] https://webdav.tuebingen.mpg.de/cause-effect/

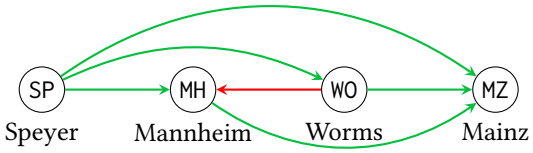

Figure 8: Results of CUTE on the *Rhein* river. A green edge represents a correct causal direction. A red edge indicates a wrong causal direction.

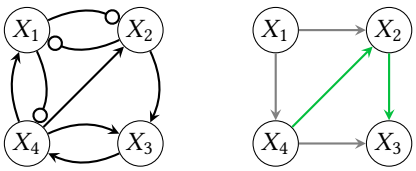

Figure 9: (left) Ground truth for the neural spike train data. A directed edge with a pointy head represents an excitatory influence, whereas a directed edge with a circular head represents an inhibitory influence. (right) Results of CUTE on the neural spike train data. A green edge represents a correct causal direction. A gray edge indicates a partially identified causal direction.

carried out on a monkey [10]. As the original experimental data itself is not available for public use, we obtained the data simulator used by the authors to confirm their experimental data. The spike trains are generated using point process generalised linear models (GLM). The values of the parameters are selected to be within the range of parameters identified in point process GLM model fits to spike trains from electrode array recording data of primate primary motor cortex [15]. Typically there are two types of influences in neural spike trains, namely excitatory (neurons fire more) and inhibitory (neurons fire less).

In Fig. 9 (left), we show the ground truth of the spike train data. We note that CUTE is not designed to deal with feedback loops. The inferred DAG after testing pairwise causality using CUTE is presented on the right side of Fig. 9. We see that it correctly infers the direction of the two non-looped causal connections, and correctly recovers one of the two causal connections (as opposed to saying there is none) for X_1 and X_2, X_1 and X_4, and X_4 and X_3 each.

When we remove the loops from the generating model by removing the inhibitory connections, we obtain the generating model depicted in Fig. 10. When we use CUTE to determine the causal direction of all dependent edges, we find that it recovers the ground truth.

5 Discussion

The experiments show that CUTE works well in practice. It reliably identifies true causal direction in a wide range of

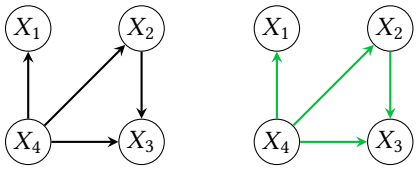

Figure 10: (left) Ground truth for the neural spike train data after removing the inhibitory influences, and cycles. (right) Results of CUTE on the modified neural spike train data. A green edge represents correct causal direction.

settings, is remarkably fast, and outperforms the state of the art by a wide margin, while qualitative case studies confirm that the results are sensible.

Although these results are promising, we see many possibilities for further improvement. We focused mainly on binary data in this work. But the extension to discrete data is also straightforward. We can compute P_{SNML} for a discrete time series by using the ML estimator relative to a multinomial family. For the conditional compression, we can trivially extend the proposal in Section 2.5.1.

In this work, we do not take into account the *instantaneous* effects. In theory, however, we can include the instantaneous effects by using the current outcome y^t of the conditioned time series y^n in conditional compression $\ell(x^n \mid y^n)$. This leads to $\ell(x^n \mid y^n) = \sum_{t=1}^{n} -\log P(x_t \mid x^{t-1}, y^t)$. We can then compute $\ell(x^n \mid y^n)$ using the technique proposed in Section 2.5.1.

Our method *might* fail in presence of a confounding time series z^n that causes both x^n and y^n. One of the avenues for future work would be to address that problem. That would also require reconsidering our assumptions, and perhaps a new framework needs to be developed with the new assumptions in mind. Last it would be interesting to explore the possibilities of using CUTE for causal discovery.

6 Conclusion

We proposed a causal inference framework for event sequences using information theory by building upon on the foundations of Granger causality. To encode the event sequences, we used minimax optimal codes relative to a parametric family of distributions. We proposed CUTE, a linear time method for inferring the causal direction between two event sequences. Extensive evaluation on synthetic, and real-world data showed that CUTE discovers meaningful causal relations in binary-valued event sequences, and outperforms the state-of-the-art.

References

[1] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[2] Y. Chen, G. Rangarajan, J. Feng, and M. Ding. Analyzing multiple nonlinear time series with extended granger causality. *Physics Letters A*, 324(1):26–35, 2004.

[3] T. Chu and C. Glymour. Search for additive nonlinear time series causal models. *JMLR*, 9:967–991, 2008.

[4] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

[5] Y. Huang and S. Kleinberg. Fast and accurate causal inference from time series data. In *Proceedings of the Twenty-Eighth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2015, Hollywood, Florida. May 18-20, 2015.*, pages 49–54, 2015.

[6] A. Hyvärinen, S. Shimizu, and P. O. Hoyer. Causal modelling combining instantaneous and lagged effects: An identifiable model based on non-gaussianity. In *ICML08*, pages 424–431. ACM, 2008.

[7] W. Kotlowski and P. Grünwald. Sequential normalized maximum likelihood in log-loss prediction. In *2012 IEEE Information Theory Workshop*, pages 547–551, 2012.

[8] J. Peters, D. Janzing, and B. Schölkopf. Causal inference on time series using restricted structural equation models. In *Advances in Neural Information Processing Systems 26*, pages 154–162, 2013.

[9] U. Pötter and H.-P. Blossfeld. Causal inference from series of events. *European Sociological Review*, 17(1):21–32, 2001.

[10] C. J. Quinn, T. P. Coleman, N. Kiyavash, and N. G. Hatsopoulos. Estimating the directed information to infer causal relationships in ensemble neural spike train recordings. *Journal of Computational Neuroscience*, 30(1):17–44, 2011.

[11] J. Rissanen and T. Roos. Conditional nml universal models. In *2007 Information Theory and Applications Workshop*, pages 337–341, 2007.

[12] J. Rissanen and M. Wax. Measures of mutual and causal dependence between two time series. *IEEE TIT*, 33(4):598–601, 1987.

[13] B. Ryabko and J. Astola. Application of data compression methods to hypothesis testing for ergodic and stationary processes. In *2005 International Conference on Analysis of Algorithms*, volume AD, pages 399–408. Discrete Mathematics and Theoretical Computer Science, 2005.

[14] T. Schreiber. Measuring information transfer. *Phys. Rev. Lett.*, 85:461–464, 2000.

[15] W. Wu and N. Hatsopoulos. Evidence against a single coordinate system representation in the motor cortex. *Experimental Brain Research*, 175(2):197–210, 2006.