Comparing Finite Sequences of Discrete Events with Non-Uniform Time Intervals

Alexander Murph[1], **Abby Flynt**[2], **and Brian R. King**[2]
[1] University of NC at Chapel Hill, 318 E Cameron Ave, Chapel Hill, NC 27599.
[2] Bucknell University, 701 Moore Ave, Lewisburg, PA 17837.

Abstract: Algorithms that quantify the similarity between two sequences of data date back to the mid 20th century. Sequence comparison continues to be active area of research in mathematics, statistics, and computer science, with applications to a wide number of fields including, biology, marketing, and linguistics. While many methods exist for comparing sequences of discrete events, this paper presents a novel method to compare such sequences while also utilizing information available from non-uniform time intervals between events. The Sequence Alignment with Non-Uniform Time Intervals (SAWNUTI) method, an extension of the Smith-Waterman and Needleman-Wunch algorithms, is described and evaluated using a simulation study and two real-world medical data sets (diabetes and eye tracking). Results illustrate the necessity of this method when time is important to consider in the comparison of sequences.
Keywords: Needleman-Wunsch, Smith-Waterman, sequence comparison, time gaps, time intervals.

1. INTRODUCTION

Methods to compare sequences have been widely researched, particularly within the bioinformatics community where biologists strive to understand ever-increasing repositories of biological sequence data. Great strides have been made in genetics and genomics, partly owed to modern sequence comparison and search tools which help researchers efficiently search large biological databases for similar sequences of interest (Lipman *et al.*, 1990). DNA, which are sequences of nucleotides represented by the symbols A, C, G, or T, represent only a small fraction of the sequence universe. For example, sequences are used to represent text, where each symbol is a letter or a word; customer transactions, where each symbol represents a potential item of purchase; or eye tracking data, where a subject's gaze is recorded while they are observing a stimulus. Sequence comparison methods can reveal interesting, frequently occurring patterns contained among similar sequences, or provide valuable information about the processes that generate similar sequences.

Sequence comparison methods have many direct applications in sequential analysis. One example of this can be seen when trying to discern anomalies over time in networks. This problem, which fits into the framework of quickest change detection (see: Zou and Veeravalli (2018); Datta *et al.* (2004); Fellouris and Sokolov (2016)), attempts to model the trajectory of an anomaly moving through a Markov Chain, where the Markov Chain is not directly observable. The only data available are noisy samples from a distribution that depends on the current (unknown) location of the anomaly in the Markov Chain, making the data a Hidden Markov Model. This problem, which has received much modern attention in mathematics and computer science (see: Zou *et al.* (2019); Rovatsos *et al.* (2017); Rovatsos *et al.* (2020)), would benefit greatly from advances in sequence comparison tools. With these tools, it would be possible to classify the anomaly

Address correspondence to A. C. Murph, Department of Statistics and Operations Research, University of NC at Chapel Hill, Chapel Hill, NC; E-mail: acmurph@live.unc.edu

itself by comparing its modeled trajectory against previous data, assuming that the movement of an anomaly through the system is dependent upon its type/cause. The algorithm introduced in this paper takes this idea a step further, introducing a way to logically account for non-constant time-intervals between movements in the system.

Many sequences are temporal, where observations in the sequence have a timestamp, and therefore an interval of time associated between observations. When the observations are uniformly recorded at fixed time intervals, standard sequence comparison methods may be used (Sankoff and Kruskal, 1983). In this case, only a start time and sample frequency are necessary to recreate the time component of the data. However, not all sequences have observations collected at regular, uniform time intervals. For example, consider employee entry and exit data in a building, item purchase patterns over the course of a day in a grocery store, drugs or other treatments given to patients in a hospital over time, or the order of links clicked by users of a website; these are among many such examples of non-uniform temporal sequence data. Non-uniform sequence data must contain a timestamp with every observation in the sequence, indicating the time when the element in the sequence occurred. While many methods exist for comparing general sequences of discrete data (Chao and Zhang, 2008; Xing *et al.*, 2010), and most of these work well on time-based sequences that are uniform, they will fail to work properly on non-uniform temporal sequence data.

Though this article focuses mostly on finite sequences of non-uniform discrete data, it is important to briefly illustrate how non-uniform time series data are compared. Time series, which are sequences of continuous data over time, have many methods available for comparing these data when samples are uniform, such as the Sequence Weighted Alignment Scoring (SWALE) model from Morse and Patel (2007) or the Spatial Assembling Distance (SpADe) from Chen *et al.* (2007). Though non-uniform time series are challenging, these series can be transformed to evenly spaced, uniform intervals through interpolation, thereby reducing the problem to one of comparing uniform time series. A common approach is the Dynamic Time Warping (DTW) algorithm from Sankoff and Kruskal (1983). In general, DTW takes two time series as input and performs a resampling of the series such that observations occur at identical time intervals, i.e., it "warps" each series in such a way that the time interval of the series matches, thus allowing any variety of time series comparison methods to be used on the warped series.

DTW methods work well for non-uniform time series of continuous data, and similar methods would work on sequences of ordinal data, but are not applicable when each observation in the non-uniform sequence is categorical. Existing methods that address non-uniform sequence comparison of discrete events are limited, and often developed to address specific types of data. For example, methods that extract trends from eye tracking data are of particular interest (Cristino *et al.*, 2010). The ScanMatch method compares saccadic eye movements by translating discrete events in time into sequences that can then be fed through a standard sequence alignment algorithm (Cristino *et al.*, 2010). The process involves binning a given subject's eye movements to create a sequence that retains the location, time, and order of information. Different possible areas on which a subject's eyes could fixate are assigned as states, which are then replicated in the sequence a number of times proportional to the length of time a subject was looking at a given area. ScanMatch, while traditionally applied to eye tracking data, can be utilized on other sequences of discrete events with non-uniform time gaps, and will serve as a useful comparison for our model.

1.1. Sequence Alignment Defined

The most common approach to compare two arbitrary finite sequences of categorical data for similarity assessment is to employ the Needleman-Wunsch (NW) or the Smith-Waterman (SW) algorithms (Needleman and Wunsch, 1970; Smith and Waterman, 1981). Originally developed to assess similarity between biological sequences, they have general applicability for any sequence of discrete observations. These methods determine similarity between two sequences by performing an alignment, where symbols in each sequence are lined up to maximize similarity. The NW and SW methods are highly similar to each other, with a small

nuance that differs the two, resulting in NW producing a global alignment, and SW a local alignment. We briefly present these methods to provide the necessary foundation for the presentation of our method.

Let $\mathscr{S}$ represent a finite set of discrete symbols (also referred to as discrete events, or categorical observations). Let $\Phi = \phi_1\phi_2\ldots\phi_m$ and $\Theta = \theta_1\theta_2\ldots\theta_n$ be sequences drawn from $\mathscr{S}$ of finite length m and n, respectively. The pairwise sequence alignment methods presented are dynamic programming techniques (Cormen *et al.*, 2009; Needleman and Wunsch, 1970; Smith and Waterman, 1981), utilizing memoization to cache intermediate results as the method progresses. Let matrix H represent an $(m+1)\times(n+1)$ matrix, serving as the cache that stores intermediate *scores* - results that are computed as the algorithm progresses. A similarity function $sim(\phi,\theta)$ must be defined that returns a relative numeric score related to the similarity of any two observations in $\mathscr{S}$. (Variations on this function are defined in each of the original definitions of the SW and NW algorithms (Needleman and Wunsch, 1970; Smith and Waterman, 1981).) This process maximizes the similarity score of Φ and Θ using the calculation $\sum_{i=1}^{\max m,n} sim(\phi_i,\theta_i)$. Notice that two elements can only be compared when they are at the same index in their respective sequences (assume the shorter sequence is padded by null values that always produce similarity score of zero). To maximize this score, the process introduces shifts in either sequence, referred to as "gaps", that are penalized by a gap penalty function. The gap penalty function $W(x)$ decreases the similarity score depending on the length of the gap introduced in the alignment process: $x \in \mathbb{N}$. There are a wide number of gap penalty functions available, with the affine being among the most common:

$$W(x) = \begin{cases} g & \text{constant gap penalty} \\ gx & \text{linear gap penalty} \\ h+gx & \text{affine gap penalty} \end{cases} \tag{1.1}$$

The NW and SW algorithms are recursive, dynamic programming algorithms that involve two phases: a scoring phase and a traceback phase. During the scoring phase, the scoring matrix H is filled with values that represent a locally optimal comparison for a given pair of subsequences. The process reuses these locally optimal values sequentially to reduce the number of calculations necessary for larger and larger subsequences. Positive values accumulate as matches occur, whereas negative values reduce the score as mismatches and gaps are introduced. The final similarity score is obtained at cell $H[m+1,n+1]$ for NW, or the maximum value cell in H for SW. Gaps and aligned pairs of observations can be output my means of the Traceback phases defined in the original definitions of these algorithms (Smith and Waterman, 1981; Needleman and Wunsch, 1970).

The sequence alignment method works by filling in the scoring matrix according to the following recursive definition:

Algorithm 1: Generalized sequence alignment (global)

Input: $\Phi = \phi_1\phi_2\ldots\phi_m$ and $\Theta = \theta_1\theta_2\ldots\theta_n$: sequences over $\mathscr{S}$
$sim(\phi,\theta)$: similarity function
$W(x)$: penalty function for gap of length $x \in \mathbb{N}$

Output: scoring matrix H

$H \longleftarrow (m+1)\times(n+1)$ matrix;
Initialize first row and column of H;
for $i \leftarrow 1...m$ **do**
  for $j \leftarrow 1...n$ **do**
    k = gap in sequence Φ;
    l = gap in sequence Θ;

$$H_{i,j} = \max \begin{cases} H_{i-1,j-1} + sim(\phi_i,\theta_j) \\ H_{i-k,j} - W(k) \\ H_{i,j-l} - W(l) \end{cases}$$

  end
end

As shown by the different gap penalty functions (Eq. (1.1)), the penalty for adding a gap in the final sequence alignment can be linearly related to the total length of that run of gaps. Whenever adding a gap, which happens when a cell of H is filled in by anything other than the value diagonal from it, k and l must be considered.

The NW and SW methods have remained the quintessential models for sequence alignment and comparison, providing the basis for many methods used today (for example, see Yates and Keedwell (2017); Prasad and Jaganathan (2018); Muflikhah *et al.* (2018)). These methods however are not suitable for non-uniform time-based sequences.

1.2. Limitations of Sequence Alignment on Non-Uniform Temporal Data

To illustrate the limitations of existing sequence alignment methods, consider the three sequences in Figure 1. Most sequence alignment methods would likely conclude that the first and third sequences are most similar. However, suppose there was a non-uniform time component to these observations, resulting in a substantial difference in the time of occurrence between A and B in the first and third sequence. In this situation, it may be more accurate to conclude that the first two sequences are most similar, because the information from the time between observations in sequence one and two causes symbols A, B, and C to better align. Moreover, if G is an observation that can be skipped to maximize the alignment, then the first two sequences will be even more similar.

This paper presents a new algorithm, the Sequence Alignment with Non-Uniform Time Interval (SAWNUTI) method, which extends the existing capabilities of two common pairwise alignment algorithms: the Needleman-Wunsch (NW) algorithm and the Smith-Waterman (SW) algorithm. These algorithms have provided an excellent framework for numerous methods of sequence comparison over the years, and adding the ability to perform an alignment of non-uniform sequences address an important, relevant need in the research community, for many domains. In this article, we introduce SAWNUTI to compare and align sequences of discrete events in time, report experimental results with a simulation study, and demonstrate the utility of the method on eye tracking and diabetes data.

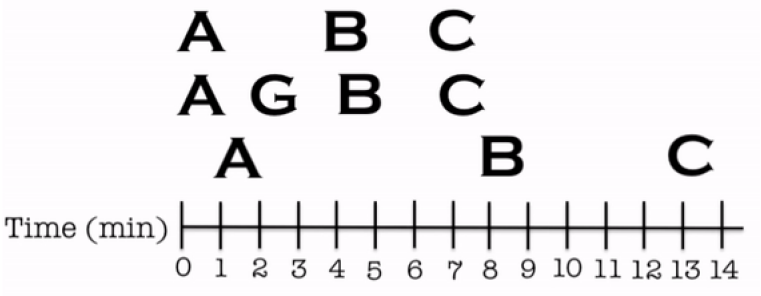

Figure 1: Sequence data with consideration of time. When time is important, one could argue that the first two sequences are most similar. When time is not important, the first and third sequences are clearly the most alike.

2. SEQUENCE ALIGNMENT WITH NON-UNIFORM TIME INTERVALS

In this section, we present the Sequence Alignment with Non-Uniform Time Interval (SAWNUTI) method to align pairs of finite-length sequences of temporal categorical data. We require each element in every sequence to be aligned to have a time stamp, and each observation is assumed to occur in chronological order.

We define our method using notation similar to the definition of the NW and SW methods in *Section 1.1*. Specifically, we let H be a $(m+1)\times(n+1)$ scoring matrix that is used to store intermediate values computed by the method. Likewise, we let *sim* be some similarity function that computes a similarity metric between two elements in sequences A and B, respectively.

We focused on modifications to the quintessential NW and SW algorithms, but expect that our modifications may be applied to a wide range of similar approaches. Like the original methods, a quantitative measure was derived which provided a metric to assess the relative similarity between two categorical sequences over time. Our method maintained the objectives of the original algorithms, where the extension of the NW algorithm performs a global alignment and the extension of the SW algorithm performs a local alignment.

2.1. Notation and Definitions

We let $\Phi = \phi_1\phi_2\ldots\phi_m$ and $\Theta = \theta_1\theta_2\ldots\theta_n$ be sequences of categorical observations of finite length m and n, respectively, maintaining consistency with notation defined in Section 1.1. Likewise, each ϕ_i and θ_j represents a single element drawn from a finite set of symbols, denoted as $\mathscr{S}$:

$$\forall 1 \le i \le m, 1 \le j \le n,\ \phi_i, \theta_j \in \mathscr{S}$$

Between each observation in every sequence, there is some interval of time. Categorical sequences Φ and Θ each have their own corresponding sequences of time intervals, denoted T (2.1) and $\mathscr{T}$ (2.2), respectively, and each pair $\phi_i, \phi_{i+1} \in \Phi$ has a corresponding time stamp t_i, and each $\theta_j, \theta_{j+1} \in \Theta$ has a corresponding time interval τ_j. There can optionally be a time interval between the first observation and whatever is considered the "start time" for the sequence. This start time is denoted as t_0 for sequence Φ and τ_0 for sequence Θ:

$$T = t_0\ t_1\ t_2\ \ldots\ t_{m-1} \tag{2.1}$$

$$\mathscr{T} = \tau_0\ \tau_1\ \tau_2\ \ldots\ \tau_{n-1} \tag{2.2}$$

Temporal sequence data is often specified with a time stamp for every observation in the sequence, rather than a time interval between all pairs of sequential observations. In this case, transforming the temporal component of the data from time stamps to time intervals is a simple subtraction of adjacent time stamps in the sequence, while retaining the first time stamp to be assigned to t_0 or τ_0, respectively. Values of time intervals are required to be sequential and consistent with the underlying time stamps, and each time interval must be greater than or equal to zero:

$$\forall 1 \leq i \leq m, 1 \leq j \leq n,\ t_i \geq 0, \tau_j \geq 0$$

Rather than convert all times to one common unit of time, time intervals are normalized using a zero-one scale transformation in order to account for varying time scales.

2.1.1. Time Interval Bias - α

There is an important trade-off that the user must make when determining how to compute an optimal alignment between temporal sequences Φ and Θ. If the user determines that only the alignment of categorical observations are important (i.e. time intervals are not to influence the final alignment), then SAWNUTI reduces to an exact implementation of the existing NW and SW methods as defined in Section 1.1. Our aim is to align sequences of categorical data with time intervals, where the user has determined that both the observations and the time intervals are to influence the final alignment. To this end, we introduce a new parameter, α into the algorithm. This parameter, called the *time interval bias*, allows the user to weight the influence of time in the sequence comparison algorithm. It is required that $\alpha \geq 0$, with larger α giving time greater influence on the final alignment. When $\alpha = 0$, the algorithm reduces to an exact implementation of SW or NW. As the value of α increases, the significance of the temporal component of the sequence alignment increases, thereby also reducing the importance of the symbols in the sequences.

The ability to determine a universal method for selecting α is still an open question. The choice of α is contingent on the research question of interest, and will often change depending on a researcher's prior knowledge of the importance of time in a study. With the example of treatments given to a patient in a hospital over time, the time at which a patient receives a treatment can be extremely important, and would likely demand a large α. Comparing items bought at a grocery store, however, may not be as time dependent, which may call for a smaller α. While we hope that good choices of α will be uncovered with more usage of SAWNUTI, we have some suggestions. First, one must choose an $\alpha \in [0, \eta)$ where $\eta = (\max_{a \in \mathscr{E}} |a|) * \left(\max_{b,c \in \mathscr{S}} sim(b,c)\right)$. $\mathscr{E}$ here is our sample space of sequences, $\mathscr{S}$ is our sample space of events, $|\cdot|$ returns the length of sequence, and $sim(b,c))$ is the similarity score of two events (discussed further in our explanation of the algorithm). This choice of α ensures that a pair of identical sequences will be given a non-negative score. While this is ultimately unnecessary for the optimization calculation, we believe this helps with the interpretability of the similarity score. Second, since there is an additional penalty parameter in our calculation (see: Section 2.1.3), it is important to consider the effects of this other parameter when choosing α. We discuss this further in Section 2.1.3.

2.1.2. Time Gap Penalty Function - $\Phi_{k,l}(i,j)$

In the original algorithm presented in *Section 1.1*, the function $W(k)$ was defined (see Eq. (1.1)) to compute the penalty for introducing a positional gap of length k as the scoring matrix H is filled. For time stamped sequences, we introduce an additional gap penalty function, which calculates a penalty based on the gaps of time that are introduced in order to maximize the aligned elements in the sequence. Function $\Phi_{k,l}(i,j)$ is known as the *time gap penalty function*, and is defined as follows:

$$\Phi_{k,l}(i,j) = |(t_i + t_{i-1} + \cdots + t_{i-l}) - (\tau_j + \tau_{j-1} + \cdots + \tau_{j-k})|. \tag{2.3}$$

Here, the values $i, j \in \mathbb{N}$ are the indices of the scoring matrix H, while the values $k, l \in \mathbb{N}$ are the lengths of the gaps in the first and second sequences, respectively. These values are equivalent to the values i, j, k, l in **Algorithm 1**.

2.1.3. Positional Gap Penalty Function - $W(x)$

The original positional gap penalty function as defined in Eq. (1.1) is retained in our algorithm. To illustrate its significance in the method, consider the sequences:

$$\mathbf{A} = A\ B$$
$$\mathbf{B} = A\ G\ B,$$

with time intervals $t_1 = 4$ for sequence A, and $\tau_1 = 2$, and $\tau_2 = 2$ for sequence B. In this instance, the alignment

$$\begin{array}{c} A - B \\ | \quad\ | \\ A\ G\ B \end{array}$$

would have a time gap penalty of

$$\Phi_{1,0}(1,2) = |t_1 - (\tau_2 + \tau_{2-1})| * \alpha = |(4) - (2+2)| * \alpha = 0$$

at the point where the gap in the first sequence is inserted. Thus, this gap would be included in the final alignment without any penalty for skipping over element G in the second sequence. Retaining the function $W(k)$ introduces a penalty for introducing gaps that result in observations being eliminated in any given alignment (such as element G in the example above.) The function $W(k)$ can be tuned to allow for these eliminations or to penalize them. Because the purpose of this paper is to focus on the inclusion of time and position simultaneously, this function must be constrained in such a way that the choice of the time interval bias α remains meaningful. To do this, we chose the constant function $W(k) = \frac{\gamma}{2} - \alpha$ where $\gamma = \max_{b,c \in \mathscr{E}} sim(b,c)$. Here, smaller values of α equate to a larger gap penalty, and visa versa. Extending $W(k)$ to a general function, and further examining the relationship between α and $W(k)$, remains an open area for research.

The variables k and l are of interest for both the time gap penalty function (2.3) as well as the positional gap penalty function (1.1). To illustrate how these variables are determined at runtime, note Figure 2. Here, the solid arrows correspond to the cell of H used to calculate a given element. The dotted arrows show the three potential cells that may be used in the calculation of $H_{i+1,j+1}$. For the cell to the left of $H_{i+1,j+1}$, we would have $k = 2, l = 1$; for the cell to the upper left diagonal of $H_{i+1,j+1}$, we would also have $k = 2, l = 1$; for the cell above $H_{i+1,j+1}$, we would have $k = 1, l = 0$. Knowing the lengths of the longest inserted gaps k and l allows us to calculate the time interval penalty between elements that are matched in the alignment algorithm. For every gap inserted, there is also another time interval that must be considered in the calculation.

2.2. The Algorithm

With the above definitions established, we can now define the algorithm.

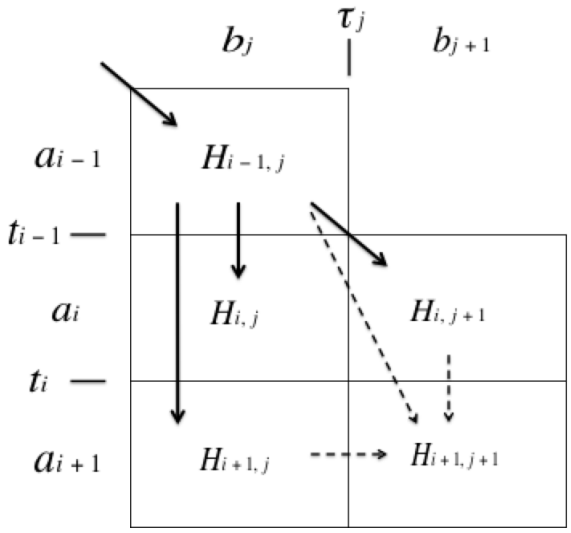

Figure 2: A subsection of the H matrix during runtime directly before the calculation of $H_{i+1,j+1}$. To determine the values of $\Phi_{k,l}(i,j), W(k)$, and $W(l)$, the length of the longest current gap in each sequence must be investigated.

Algorithm 2: Sequence Alignment with Non-Uniform Time Intervals

Input: $\Phi = \phi_1\phi_2\ldots\phi_m$ and $\Theta = \theta_1\theta_2\ldots\theta_n$: sequences over $\mathscr{S}$
$T = t_0\ t_1\ \ldots\ t_{m-1}$ and $\mathscr{T} = \tau_0\ \tau_1\ \ldots\ \tau_{n-1}$: time intervals for Φ and Θ, respectively
$\mathbb{T}$: Every time interval in the entire data set of interest
α : time interval bias
$sim(\phi,\theta)$: similarity function
$W(x)$: penalty function for gap of length $x \in \mathbb{N}$

Output: scoring matrix H

$H \longleftarrow (m+1)\times(n+1)$ matrix;
Initialize first row and column of H;
Transform each sequence into a sequence of time intervals;
Perform the zero-one transformation: $\forall \hat{t} \in \mathbb{T}$, $\frac{\hat{t}-min(\mathbb{T})}{max(\mathbb{T})-min(\mathbb{T})}$;
for $i \leftarrow 1...m$ **do**
  for $j \leftarrow 1...n$ **do**
    k = gap in sequence Φ;
    l = gap in sequence Θ;
    $\Phi_{k,l}(i,j) = |(t_i + t_{i-1} + \cdots + t_{i-k}) - (\tau_j + \tau_{j-1} + \cdots + \tau_{j-l})|$
    if *Global Alignment* **then**

$$H_{i,j} = \max\begin{cases} H_{i-1,j-1} - \Phi_{0,0}(i-1,j-1)*\alpha + sim(\phi_i,\theta_j) \\ H_{i-k,j} - \Phi_{k,l}(i-1,j)*\alpha - W(k) \\ H_{i,j-l} - \Phi_{k,l}(i,j-1)*\alpha - W(l) \end{cases}$$

    else

$$H_{i,j} = \max\begin{cases} H_{i-1,j-1} - \Phi_{0,0}(i-1,j-1)*\alpha + sim(\phi_i,\theta_j) \\ H_{i-k,j} - \Phi_{k,l}(i-1,j)*\alpha - W(k) \\ H_{i,j-l} - \Phi_{k,l}(i,j-1)*\alpha - W(l) \\ 0 \end{cases}$$

    end
  end
end

2.3. Comparative Metrics

The output of our method is a scoring matrix H. The final similarity score between two arbitrary sequences is obtained by returning the maximum value in H for local alignment, and $H_{m,n}$ for global alignment. The similarity score between a pair of sequences can be interpreted as a distance metric. For studies examining exhaustive pairwise sequence comparisons over more than two sequences, distance matrices may be formed to store all pairs of comparisons over a set of sequences. The resulting distance matrix is passed to a hierarchical clustering algorithm (with the `hclust` package in `R` (R Core Team, 2015)) using Ward's linkage (Maimon and Rokach, 2010). The clustering can be used to generate a dendrogram as a means of visualizing groupings of sequences which exhibited high similarity.

From the clustering, we can classify similar sequences into groups and compare that grouping to known class labels using the Adjusted Rand Index (ARI) (Rand, 1971). The ARI is a numerical measure of agreement between any two groups of data, where a value of 1 indicates perfect agreement and under random partitioning, the expected value of the ARI is zero. The ARI assesses how accurately the SAWNUTI method is able to use variation in sequence events and time between events to quantify the similarity of sequence

data. The ARI is used only in experiments where known class labels exist.

3. DATA

To demonstrate the efficacy and accuracy of the algorithm, we developed a method to simulate categorical sequences with non-uniform time intervals using Markov models. Additionally, we used two real-world datasets to assess the method. This section provides descriptions and any necessary pre-processing steps for the different data.

3.1. Simulated Categorical Sequences with Non-Uniform Time Intervals

We simulate a set of non-uniform categorical sequences using a Markov model, which is a stochastic process that assumes the Markov property. That is, consider a system that can be in any of one of a finite number of states, from a set of states $\mathscr{L}$, observed at the discrete moments in time $n = 0, 1, 2, \ldots$. Then, let the random variable X_n denote the state of the system at time n, the Markov property states that for $n \geq 0$:

$$P(X_{n+1} = x_{n+1} | X_0 = x_0, \ldots, X_n = x_n) = P(X_{n+1} = x_{n+1} | X_n = x_n),$$

for the states $x_0, \ldots, x_{n+1} \in \mathscr{L}$ (Hoel *et al.*, 1972).

For the Markov models used in this simulation, we define the possible states in our system as $\mathscr{L} = \{A, B, C, D, E\}$. All states have a non-zero probability of moving to any other state, including itself. To simulate a sequence of length $n > 0$, we simply record sequentially in what state our Markov model was at the times $0, 1, 2, \ldots, n-1$, with the initial state being chosen at random. For the purposes of this simulation study, we chose a random n from the range $30 \leq n \leq 40$ for each new sequence produced. This range was chosen to be similar to the sequence lengths found in the real data used in this paper.

Two Markov models, named "Model A" and "Model B", were created for this simulation. All transition probabilities for each of these models can be found in Tables 1 & 2 in the Appendix.

The process described for using Markov models created non-temporal sequences of discrete events. To simulate the required (non-uniform) time intervals, we drew values from a Uniform distribution. For any given sequence of length $l \geq 0$, we drew l values from Uniform$(10, u)$. The upper bound u was tuned to experiment with different time variations. For a value of u closer to 10 (but still greater than 10), there was little variation in the different time intervals. In contrast, a greater u meant a wider variation. For the purposes of this simulation, we chose two values for u: 20 and 50. We considered sequences simulated with $u = 20$ to have "narrow" time intervals while sequences simulated with $u = 50$ to have "wide" time intervals.

The data simulation involved producing 20 sequences: 10 sequences from Model A and 10 sequences from Model B. Of the sequences created from Model A, 5 were given time intervals with a $u = 20$ and 5 were given time intervals with a $u = 50$. The same split of time intervals were given to the sequences from Model B. To account for random variation in the simulation, we analyzed 10 replications of the 20 sequences (discussed further in Section 4.1).

3.2. Diabetes Data

We obtained a medical dataset of diabetes patients who were logging their condition and their behavior related to managing their diabetes over time. Each sequence in this dataset corresponds to a single patient's record of events over time, where possible events include things like insulin doses, glucose measurements, meals, and exercise. This is a benchmark machine learning data set obtained from the UCI ML repository (`https://archive.ics.uci.edu/ml/datasets/diabetes`). Patient records contained data recorded from an automatic electronic recording device and from transcribed paper records. The automatic device had an internal clock to record precise time stamps. However, paper records only provided relative,

general times, such as breakfast, lunch, dinner, or bedtime. Paper records were transcribed to generally accepted time stamps, with breakfast = 08:00, lunch = 12:00, dinner = 18:00, bedtime = 22:00. Thus paper records have approximate uniform recording times whereas electronic records have more realistic time stamps. Both of these data sources were combined into a single file for each patient.

To limit any differences caused by time of year, we obtained sequence data on 14 patients that all had measurements from the week August 19 - August 21, 1990. One patient with fewer than 10 measurements was removed, since we would expect this patient to be distinct from the other patients just by merit of having too few observations. This left us with 13 sequences to compare, ranging in length from 17-33 recorded events.

3.3. Eye Tracking Data

Eye tracking systems record the gaze of an individual as they observe a stimulus over a short, fixed period of time. These devices record raw gaze points at a fixed time frequency, where each point is stored as an x-y coordinate and a time stamp. (Numerous methods exist for collecting and analyzing eye tracking data. We recommend Duchowski and Duchowski (2003) for the interested reader.) These data are usually transformed into a non-uniform sequence of *fixations*, which are locations on the stimulus that capture the attention of the subject for some period of time. An individual fixation is comprised of a x-y coordinate, a time stamp indicating when the fixation started during the experiment, and a duration, indicating how long the subject's gaze was on this location.

We evaluated our method on an eye tracking dataset provided by the Geisinger Autism and Developmental Medicine Institute (Lewisburg, PA) from a study that collected eye tracking data for 64 participants using a Tobii 60XL eye tracking system. Each participant was presented with an array of objects arranged across a single image (Figure 3) for 10 seconds. (This image is one of many that have been used in numerous studies in autism (Sasson *et al.*, 2008; Elison *et al.*, 2012).) The data presented were cleaned, filtered, and post processed by Tobii software to identify significant fixations for each participant. The data received were anonymized, with each participant represented as a single random numeric identifier along with their unique sequence of non-uniform fixations. Fixations were mapped to their underlying *areas of interest*, which are areas on the underlying stimulus that are expected to capture the attention of the participant. Each area of interest was assigned to a unique identifier, and thus original fixation sequences were transformed to sequences of area of interest identifiers. Finally, adjacent identical identifiers in each sequence were merged. Figure 3 shows an example scanpath fully processed and a time line of these fixations over the 10 second period of the experiment.

4. EVALUATION

In this section, we present the results of the SAWNUTI method on the simulated data, as well as both real-world datasets. The implementation for the SAWNUTI method and the sequence simulation were developed in the `R` programming language. All code will be made available on GitHub upon publication. For simplicity and to allow us to focus on the time bias parameter, for all data, the positional gap penalty function $W(x) = 2, \forall x \in \mathbb{N}$.

4.1. Results on Simulated Data

The primary aim of this simulation study was to investigate how the introduction of time affects our ability to distinguish between sequences from different known groups. To test the effects of time, we varied the time interval bias α and reviewed how this variation affected the comparative metrics. For each $\alpha \in \{0, 0.5, 1.0, \ldots, 4.5, 5.0\}$, we clustered the sequences and obtained two groups using the similarity scores

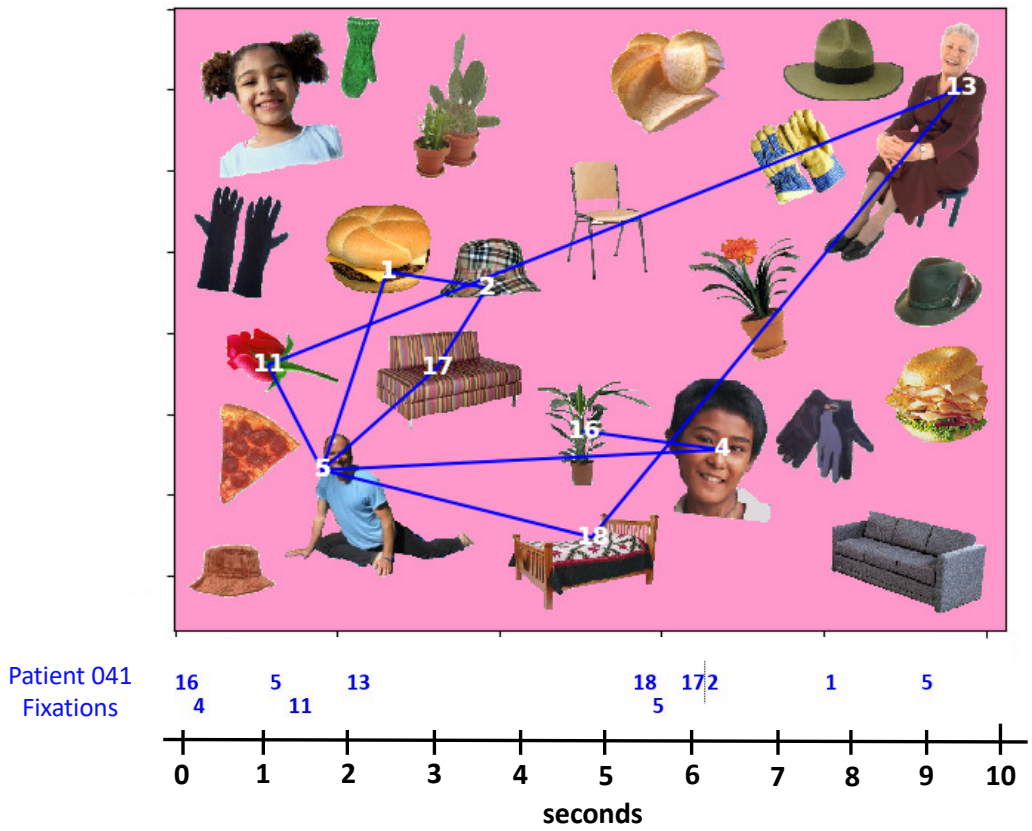

Figure 3: An example scan path for a single participant is plotted on the stimulus used in the study. Labels indicated along the path are area of interest identifiers. The bottom expresses the times at which the participant fixated on each area of interest. The spacing between each identifier is proportional to the length of the time interval between those events. Events occurring simultaneously (or nearly simultaneously) with the events preceding them are stacked downward to prevent overlap in the visual.

outputted by the SAWNUTI algorithm (see Section 2.3 for an explanation of the clustering process used). We then compared the groups discovered from our method against the known groups, denoted "Model *A*", "Model *B*", as well as against the time interval groups "narrow intervals ($u = 20$)" and "wide intervals ($u = 50$)". This produced two ARI values, which we used as our comparative metrics for this study: the ARI for the *model* classifier and the ARI for the *time interval* classifier.

The choice of $\{0, 0.5, 1.0, \ldots, 4.5, 5.0\}$ as our set of possible α values was based on a number of things. We began the range at zero because we wanted to see what would happen when time had no effect on the calculation whatsoever. We then chose a range of α values in relatively small increments until we had an α size much greater than (here, more than double) the value of our gap penalty. The goal of this simulation is to see how having an α well above, and well below, the gap penalty score effects the overall calculation, and corresponding clustering. Again, a theoretical means to chose α does not exist because it completely contingent on the research question of interest. Indeed, the "right" value of α for a given study depends on how interested in time the researcher is, and may change across different studies even when they use identical datasets. It is for this reason that a simulation study is necessary, despite the limitations inherent in picking parameter settings without theoretical justification.

As described previously, to account for variation caused by a particular random draw of our 20 simulated sequences, we performed 10 replications of this entire process; producing a total of 100 ARI values: 10 ARIs for each of the possible 10 values of α. This allowed for the distribution of ARI values to be compared across different α.

While both the global and local SAWNUTI algorithms were accessed with the simulation study, we will only present the results for the global algorithm. Results for the local algorithm can be found in the Appendix.

The side-by-side box plots in figure 4 show changes in ARIs for the global SAWNUTI algorithm as the clusters were compared against the known groups "Model *A*" and "Model *B*" and against the known groups "narrow intervals" and "wide intervals". For the SAWNUTI method, lower values of α (which focused solely on the variations in sequence elements) gave better ARIs when calculated using the model labels "Model *A*" and "Model *B*". This is evidenced in figure 4, where the ARIs that express the algorithm's ability to distinguish the Markov model of origin are higher for low values of α and lower for $\alpha > 2.5$. Thus, when the time interval bias is increased, the entire distribution of ARIs decreased as time became more important to the alignment of the sequences. Conversely, a greater time interval bias α translated to greater ARI values when those ARIs were calculated using the time interval range groupings "narrow" and "wide". This is evidenced by an opposite trend for the ARIs that express the algorithm's ability to distinguish the time interval range of origin: as the time interval bias α increases, the distribution of ARIs at each α also increased towards 1 (perfect classification).

Overall the simulations by means of Markov models exhibited the desired attributes in the proposed algorithms. As the time parameter was increased, the SAWNUTI algorithm began to classify sequences based off of the type of variation in their time parameters. When time was taken out of the equation by lowering α, SAWNUTI performed the same tasks as the NW and SW algorithm: grouping observations using only sequence element variation.

To contrast the performance of the SAWNUTI method on this particular set of simulated data, we implemented the ScanMatch method in the `R` programming language (Cristino *et al.*, 2010). ScanMatch utilizes a temporal binning technique that repeats states in a sequence based on the length of time a given patient looks at that state. The method then compares the modified sequence using the NW algorithm. In much the same way we varied α with the SAWNUTI method, a multiplier β was varied in ScanMatch, allowing us to control the number of states added into the sequence for a fixed length of time. For our implementation, a given β translates to

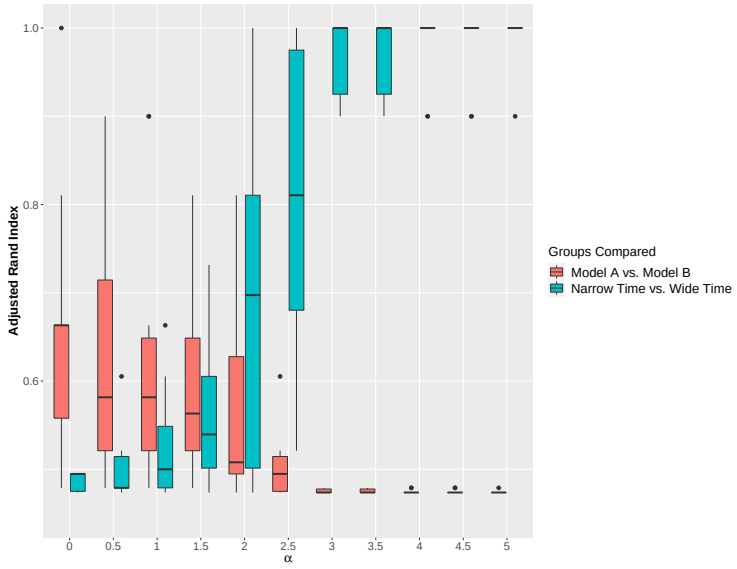

Figure 4: The distribution of ARIs obtained for different values of the time interval bias α. Using our extension of NW algorithm, we attempt to distinguish a sequence's *Markov model* and *time distribution* of origin. The downward trend of the box plots for "Markov A vs. Markov B" shows how the Markov model of origin becomes less important as time is weighed more heavily, while the upward trend of the box plots for "Narrow Time vs. Wide Time" shows how the time distribution of origin becomes more important as time is weighed more heavily.

$$\left\lfloor \beta * \frac{\hat{t} - min(\mathbb{T})}{max(\mathbb{T}) - min(\mathbb{T})} \right\rfloor$$

repetitions of a state for any given $\hat{t} \in T \cup \mathscr{T}$. We tested $\beta \in \{0,1,2,\ldots,10\}$, for all 10 repetitions of the simulation study, resulting in 100 ARI values, visualized with side-by-side box plots.

As seen in figure 5, increasing the value for β caused a downward trend in the distribution of ARI values when comparing to the originating model labels "Model A" and "Model B". This parallels the effect observed with the increase of α in the SAWNUTI algorithm with these same classifiers. However, in contrast to the results observed by SAWNUTI on the "narrow" and "wide" groupings ScanMatch, showed no apparent trend with an increase in β.

Thus, the SAWNUTI method was able to utilize time appropriately to distinguish between the time variation in different sequences, while the ScanMatch method was unable to distinguish between the "wide" and "narrow" groups, regardless of the value chosen for β. It is worth noting that ScanMatch's runtime increases depending on the variation in the time gaps and chosen β. In contrast, the runtime of the SAWNUTI algorithm is only contingent upon the location and order, rather than the location, order, and time of the sequence elements.

4.2. Results on Diabetes Data

To analyze the diabetes data during a three day period in August, we calculated all possible pairwise comparisons using the SAWNUTI method for the time interval biases $\alpha \in \{0,5\}$ and the match function:

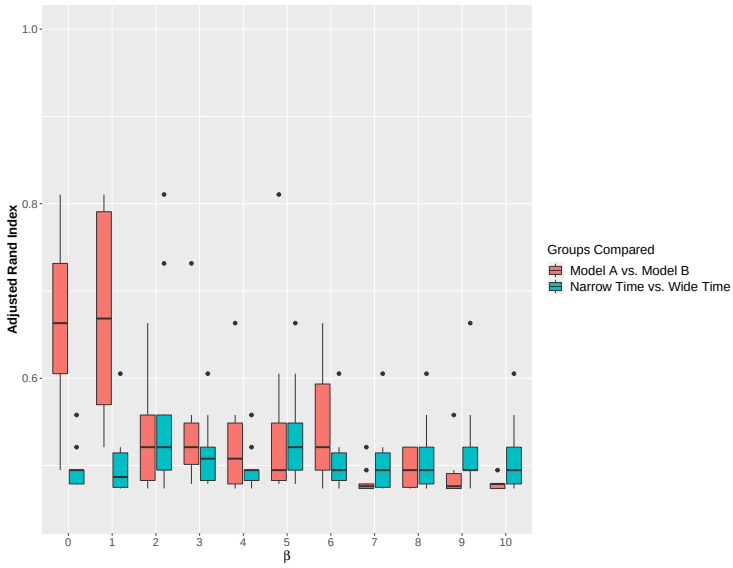

Figure 5: The distribution of ARIs obtained for different values of the binning proportion β when the ScanMatch attempts to distinguish a sequence's *Markov model* and *time distribution*. The downward trend of the box plots for "Markov A vs. Markov B" shows how the Markov model of origin becomes less important as time is weighed more heavily. The box plots for "Narrow Time vs. Wide Time" show no apparent trend in the ARI values for a higher values of β.

$$sim(a,b) = \begin{cases} 0.5 & \text{if } a = b \\ -0.5 & \text{if } a \neq b \end{cases}.$$

Since certain measurements, such as a pre-breakfast blood glucose measurement, should occur for most patients, we choose a larger α and smaller match score to give time a higher weight. Note that for this study, there were no known labels, so we therefore analyze cluster behavior without ARI measures.

Performing hierarchical clustering on the similarity scores for the 13 diabetes subjects, we saw a change in the resulting clusterings between $\alpha = 0$ and $\alpha = 5$ (figure 6). In these two clusterings, we noticed that sequences 13 and 19 were less similar to each other when $\alpha = 0$, but very similar to each other with the inclusion of time ($\alpha = 5$). To examine these two sequences further, we put them on a time scale, where the space between each event is proportional to the actual amount of time between each event (see figure 7).

The visualization aided in our understanding of why these two sequences were more alike with the presence of the time interval bias. For each sequence, there was a group of elements: [58 (pre-breakfast blood glucose measurement), 33 (NPH insulin dose), 34 (UltraLente insulin dose)], that occurred around the same time each day. There are additionally other common events that are better matched by use of this approach, such as event 60 (pre-lunch blood glucose measurement) on Day 3, and the group [64 (pre-snack blood glucose measurement), 34 (NPH insulin dose)] at the end of Day 3. These events in each sequence occur around the same time, yet would be out of alignment without including time in the algorithm, because of the difference in the number of events that occur in the overall sequences.

Without time, these two patients were not as similar. The patient represented by sequence 19 has numerous events that suggest this patient is more proactive with their disease, such as event 70 (above average exercise). The inclusion of time revealed a nuance in the similarity of these two sequences, which had similar groupings that occurred at the same time, even if many of their other observations differed.

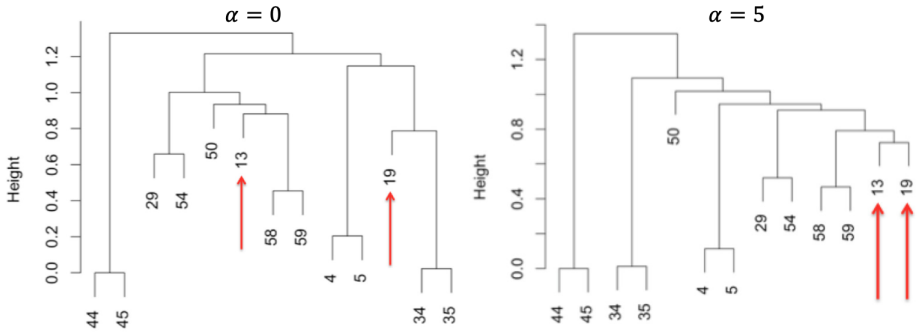

Figure 6: Dendrogram for the hierarchical clustering of diabetes subjects using the global SAWNUTI algorithm with $\alpha = 0$ and $\alpha = 5$. In the former, sequences 13 and 19 are not found to be similar, while in the latter, sequences 13 and 19 are highly similar.

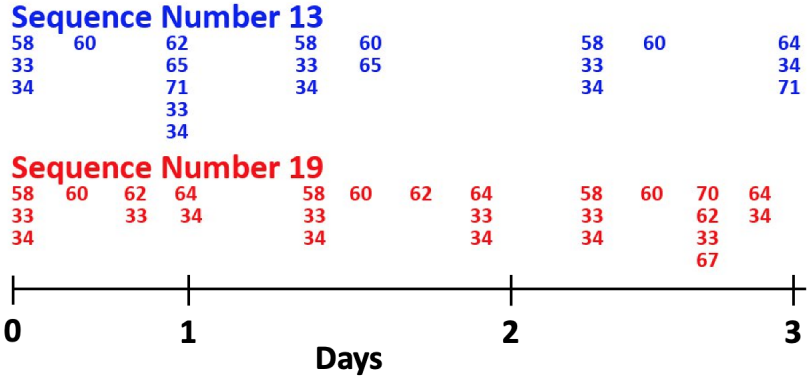

Figure 7: Sequence 13 and Sequence 19 from the diabetes data on the time axis. The space between each numerical event is proportional to the time interval between those events. Events occurring simultaneously (or nearly simultaneously) with the events preceding them are stacked downward to prevent overlap in the visual.

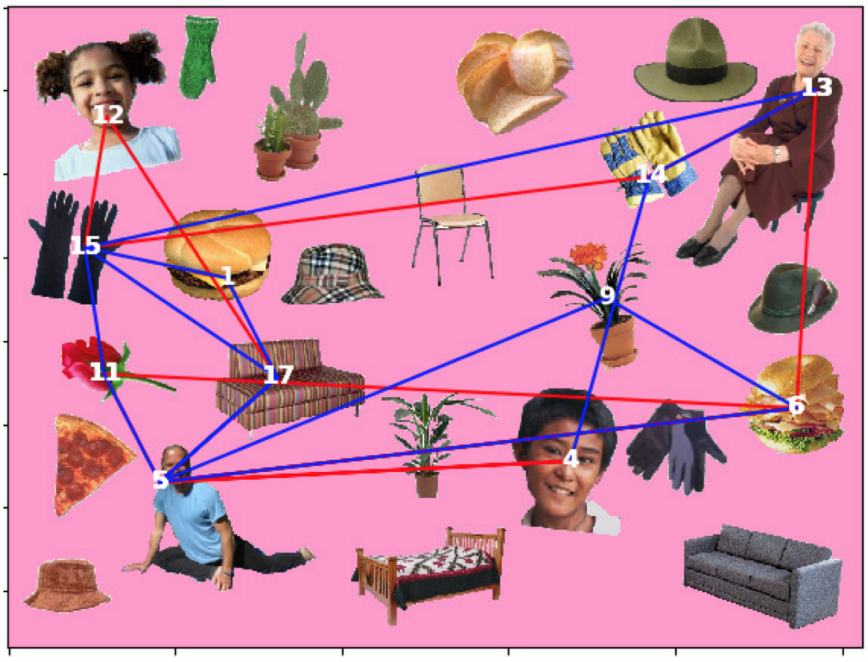

Figure 8: The trajectory of Patient 421 (in red) and Patient 261 (in blue) on the image given to the eye-tracking study patients.

4.3. Results on Eye Tracking Data

Unlike the diabetes data, the eye tracking data from Geisinger Medical Center had matches less frequently between its sequences, and so we chose a greater match score, by using the following similarity function:

$$sim(a,b) = \begin{cases} 3 & \text{if } a = b \\ -3 & \text{if } a \neq b \end{cases}.$$

With a higher match score, we also used a larger α, to ensure that the effects of α would not be rendered insignificant by the similarity function. We calculated all possible pairwise comparisons using the global SAWNUTI algorithm for the time interval biases $\alpha \in \{0, 10\}$

After calculating all of the pairwise comparisons of the available sequence observations for each α, there was a noticeable change in the similarity of the sequences for Patient 261 & Patient 421. To investigate the reason for this, we mapped these sequences onto the stimulus used in figure 8. This figure reveals where and in what order each of these two patients looked during a 10 second interval of observing the image.

To examine the effects of time on the similarity calculation of these two sequences, we perform an alignment with both $\alpha = 0$ and $\alpha = 10$, and used the traceback phase from the NW algorithm to determine which elements were being aligned for each calculation. In figure 9, we drew out all the elements on the time axis, then connected the elements matched for an $\alpha = 0$ (which is equivalent to a basic NW sequence alignment). It can be seen that the elements matched vary greatly in their time of occurrence, some by as much as 3.5 seconds (in a 10 second window). To contrast this, we created the same figure, except the matches drawn are those derived from the SAWNUTI algorithm and an $\alpha = 10$. Here, the difference in the times of occurrence for the matches is significantly less than what we saw in figure 9, when $\alpha = 0$ and time was not considered.

An increase of α affected the output of the SAWNUTI algorithm. With greater α, the method placed

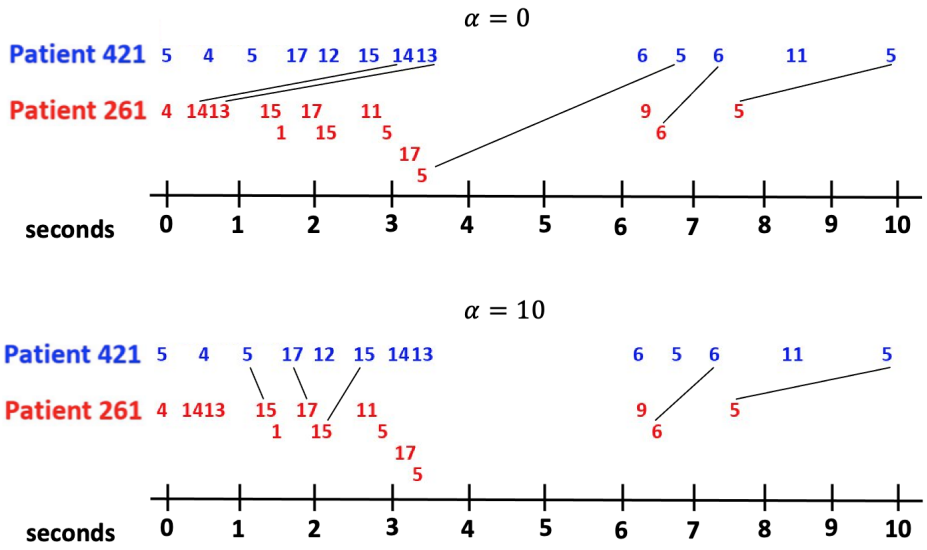

Figure 9: The alignment performed using the SAWNUTI algorithm using $\alpha = 0, 10$. Matched elements are connected by the black lines. When time is not a factor, the matches that occur vary greatly according to when they happened in time. When time is a factor, the matches (and mismatches) that occur are closer to each other in time.

a higher priority on matching fixations that are not only identical, but also occurring at similar times. In situations where numerous participants are exhibiting highly identical fixation patterns, considering the time of fixations in the similarity assessment provides a good mechanism for further distinguishing gaze patterns.

5. DISCUSSION

We evaluated the SAWNUTI method with a simulation study that compared finite sequences of discrete events in non-uniform time that were generated using Markov models, with Uniform distributions specified for the time intervals between observations. The results illustrated that our method was able to distinguish between sequences based on the variation of their time gaps. Moreover, we demonstrated that there was no loss of basic alignment functionality, as we were also able to distinguish variations in the observations within the sequence. The output produced demonstrated how the tuning of the time interval bias α, and selection of an appropriate similarity function, allows a researcher to weigh the significance of variations in time against the significance of variations in the observations.

The introduction of a time interval bias, α, in the algorithm allowed for time to be weighted differently depending on its relative importance to the application. An α of zero causes the extensions to reduce to the original SW and NW algorithms, while larger values of α make the variation in the time intervals more important in the similarity calculations, than the variation in the actual sequence elements. The values of α were chosen to reflect how heavily we wanted to weight the importance of time in the sequence alignment. In the simulation study (*Section 4.1*), we demonstrated the sensitivity of the results to changes in α, and these changes were specific to the application/data used. Future work will explore methods for choosing the optimal value of α, that can supplement the researcher's intentions and application knowledge. In conjunction with this optimization, a method for optimizing the similarity and gap penalty functions would be relevant for any choice for α.

The SAWNUTI method was illustrated on two real-world data sets from the medical domain. The di-

abetes data is a benchmark sequence dataset where each sequence represents a series of diabetes-related medical events, with some events representing individual patient behaviors, and others recorded automatically. These data demonstrated how one might be able to identify behaviors between patients, in order to identify patient medical patterns that yield similar patient outcomes.

The SAWNUTI method applied to eye tracking data showed great potential for identifying common eye scans among a pool of participants in any study involving eye tracking systems. Eye tracking systems are widely used in many industries today, including research in psychology, human-computer interaction, marketing, among many others. Regardless of the domain, eye tracking data are almost always reduced to non-uniform sequences of fixations for analysis, which are suitable for analysis with the SAWNUTI method. We showed that in taking both time and fixations into consideration, our method was able to identify groups of patients with similar gaze patterns.

Since this method involves extending the SW and NW algorithms, two quintessential sequence comparison algorithms, there has already been a vast amount of work that was built off of these algorithms that may benefit in improved alignments by utilizing time information. For example, consider BLAST, which is an approximation to these algorithms, utilized to decrease the runtime and handle large quantities of data Lipman *et al.* (1990). An extension of BLAST to include information from non-uniform time intervals would allow SAWNUTI to be applied to relatively large sequence data sets, and would require a way to update the initial preprocessing phase to consider times.

6. CONCLUSION

The SAWNUTI method was presented as an extension of the SW and NW sequence comparison algorithms to incorporate information provided by non-uniform time intervals between discrete event sequences. This method was evaluated on simulated sequences generated from Markov models that were adapted to output sequences with non-uniform time intervals. We also demonstrated the method on a diabetes benchmark sequence dataset, and on eye tracking data. The SAWNUTI method will be a valuable tool for researchers collecting non-uniform sequences of categorical observations who want to ensure that time is an important component to be used in the sequence comparison.

Data Availability and Acknowledgements
We express our gratitude to Dr. Vanessa Troiani and Dr. Antoinette Dicrisco at Geisinger Autism and Developmental Medicine Institute in Lewisburg, PA, for their interest in our work, for their insight, and for contributing eye tracking data. At this time, this data is not available to the public.

We would also like to express our thanks for the helpful suggestions from Nitis Mukhopadhyay and the tireless efforts of our reviewers.

7. APPENDIX

	A	B	C	D	E
A	0.01	0.1	0.04	0.8	0.05
B	0.18	0.01	0.11	0.1	0.6
C	0.1	0.6	0.1	0.1	0.1
D	0.05	0.8	0	0.1	0.05
E	0.8	0.08	0	0.02	0.1

Table 1: The transition probabilities for Markov model A, organized in table Δ_A. Given the cell indexed at row q and column r for $q,r \in \{A,B,C,D,E\}$, $\Delta_A(q,r) = P(X_{n+1} = r | X_n = q)$.

	A	B	C	D	E
A	0.01	0.1	0.24	0.5	0.15
B	0.08	0.01	0.01	0.1	0.8
C	0	0.6	0.1	0.1	0.2
D	0.05	0.5	0.1	0.3	0.05
E	0.6	0.18	0.1	0.02	0.1

Table 2: The transition probabilities for Markov model A, organized in table Δ_B. Given the cell indexed at row q and column r for $q,r \in \{A,B,C,D,E\}$, $\Delta_B(q,r) = P(X_{n+1} = r | X_n = q)$.

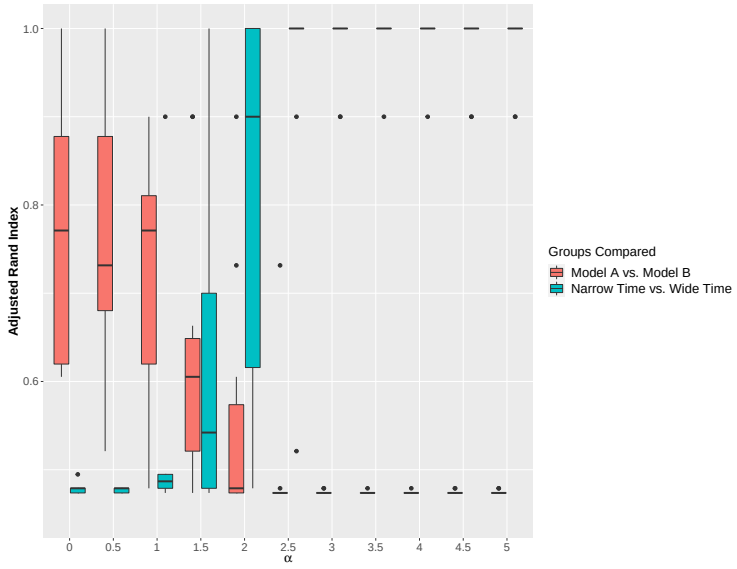

Figure 10: The distribution of ARIs obtained for different values of the time interval bias α. Using our extension of SW algorithm, we attempt to distinguish a sequence's *Markov model* and *time distribution* of origin. The downward trend of the box plots for "Markov A vs. Markov B" shows how the Markov model of origin becomes less important as time is weighed more heavily, while the upward trend of the box plots for "Narrow Time vs. Wide Time" shows how the time distribution of origin becomes more important as time is weighed more heavily.

This appendix contains results of the simulation study for the *local* SAWNUTI algorithm. Figure 10 shows changes in ARI as the clusters are compared against the known model labels "Model *A*" and "Model *B*" and shows changes in ARI as the clusters are compared against the known interval labels "narrow intervals" and "wide intervals". The results of the local SAWNUTI algorithm are similar to those of the global SAWNUTI algorithm described in Section 4.1.

REFERENCES

Chao, K.-M. and Zhang, L. (2008) *Sequence Comparison: Theory and Methods*. Springer Publishing Company, Incorporated, 1st edn.

Chen, Y., Nascimento, M. A., Ooi, B. C. and Tung, A. K. H. (2007) Spade: On shape-based pattern detection in streaming time series. In *2007 IEEE 23rd International Conference on Data Engineering*, 786–795.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2009) *Introduction to Algorithms*. The MIT Press, 3 edn.

Cristino, F., Mathot, S., Theeuwes, J. and Gilchrist, I. D. (2010) Scanmatch: A novel method for comparing fixation sequences. *Behavior Research Methods*, **42**, 692–700.

Datta, S., Chattopadhyay, S. and Mukhopadhyay, N. (eds.) (2004) *Change-Point Detection in Multichannel and Distributed Systems*, 339–370. New York, New York, USA: Dekker.

Duchowski, A. T. and Duchowski, A. T. (2003) *Eye Tracking Techniques*. Springer.

Elison, J. T., Sasson, N. J., Turner-Brown, L. M., Dichter, G. S. and Bodfish, J. W. (2012) Age trends in visual exploration of social and nonsocial information in children with autism. *Research in Autism Spectrum Disorders*, **6**, 842–851.

Fellouris, G. and Sokolov, G. (2016) Second-order asymptotic optimality in multisensor sequential change detection. *IEEE Transactions on Information Theory*, **62**, 3662–3675.

Hoel, P. G., Port, S. C. and Stone, C. J. (1972) *Introduction to Stochastic Processes*. Boston: Houghton Mifflin Company.

Lipman, D. J., Gish, W., Miller, W. and Myers, E. W. (1990) Basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403–410.

Maimon, O. and Rokach, L. (eds.) (2010) *A survey of Clustering Algorithms*, chap. 14, 269–298. Springer, 2 edn.

Morse, M. D. and Patel, J. M. (2007) An efficient and accurate method for evaluating time series similarity. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, 569–580. New York, NY, USA: ACM.

Muflikhah, L. *et al.* (2018) An improved needleman-wunsch algorithm for pairwise sequence alignment of protein-albumin. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, **10**, 83–87.

Needleman, S. B. and Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, **48**, 443 – 453.

Prasad, D. V. and Jaganathan, S. (2018) Improving the performance of smith–waterman sequence algorithm on gpu using shared memory for biological protein sequences. *Cluster Computing*, 1–10.

R Core Team (2015) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Rand, W. M. (1971) Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, **66**, 846–850.

Rovatsos, G., Zou, S. and Veeravalli, V. (2020) Sequential algorithms for moving anamaly detection in networks. *Sequential Analysis*, **39**, 6–31.

Rovatsos, G., Zou, S. and Veeravalli, V. V. (2017) Quickest change detection under transient dynamics. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4785–4789.

Sankoff, D. and Kruskal, J. B. (1983) *Time Warps, String Edits, and Macromolecules : The Theory and Practice of Sequence Comparison*, 130–150. Reading, Massachusetts: Addison-Wesley Publishing Company.

Sasson, N. J., Turner-Brown, L. M., Holtzclaw, T. N., Lam, K. and Bodfish, J. W. (2008) Children with autism demonstrate circumscribed attention during passive viewing of complex social and nonsocial picture arrays. *Autism Research*, **1**, 31–42.

Smith, T. F. and Waterman, M. S. (1981) Identification of common molecular subsequences. *Journal of Molecular Biology*, 195–197.

Xing, Z., Pei, J. and Keogh, E. (2010) A brief survey on sequence classification. *SIGKDD Explor. Newsl.*, **12**, 40–48.

Yates, W. and Keedwell, E. (2017) Clustering of hyper-heuristic selections using the smith-waterman algorithm for offline learning. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 119–120. ACM.

Zou, S., Fellouris, G. and Veeravalli, V. (2019) Quickest change detection under transient dynamics: Theory and asymptotic analysis. *IEEE Transactions on Information Theory*, **65**, 1397–1412.

Zou, S. and Veeravalli, V. V. (2018) Quickest detection of dynamic events in sensor networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6907–6911.