Application of a Novel Fuzzy Pattern Mining Algorithm for Sequence Data

***Abstract*—For many Markov chains that arise in applications (network security, health, finance, etc.), state spaces are huge, and existing matrix methods may not be practical or even not possible to implement. For example, the heart of Google's search engine is the PageRank algorithm based on Markov Chain, which assigns an importance value to each webpage for over one trillion web pages. In the literature, the expected waiting time for Markov Chain (with a smaller number of states) generated patterns is obtained by finding an appropriate pattern matrix and solving a set of linear equations. In this paper, first, a novel fuzzy transition probability matrix is introduced, and a novel pattern mining algorithm is proposed for sequence data of any length. The driving idea, unlike the existing work, is that the resilient pattern mining algorithm is obtained by fuzzifying the transition probability matrix. The proposed algorithm, which avoids the inversion of the pattern matrix, is applicable to Markov chains with huge state spaces. Three examples (DNA sequence data (3954 base pairs), patterns generated by the log returns of the stocks/cryptocurrencies, and Markov chain model for network security) of the application of the proposed algorithm are discussed in detail. The main contribution of this paper is to fit an appropriate Markov chain model to a given sequence data and use the proposed fuzzy pattern mining algorithm to obtain resilient probabilistic forecasts and expected waiting time for patterns of interest. Often, the concept of scaling is implemented to extract relevant information for sequence data. Optimal scaling and spectral envelope leading to efficient estimates are also discussed in some detail.**

***Index Terms*—Markov chains, Sequence data, Fuzzy Stochastic Matrices, Pattern Mining Algorithm, Directional Forecasts, Optimal Waiting Time**

I. Introduction

Recently, there is a growing interest in using Markov chain models in studying patterns in sequence data. Some well known examples of sequence data are DNA sequence and trading signals generated by the stock/cryptocurrency prices. In many practical applications, first, a Markov chain model is fitted to the sequence data and the transition probability matrix is calculated. Once the sequence data are presented, researchers are more interested in particularly Markov chain generated patterns. The process of investigating and identifying underlying processes for specified patterns is known as pattern mining. The purpose of pattern mining in large data sets is to capture and compare trends that have previously been associated with observations such as diseases, trading, malicious events in DNA sequence, finance and computer network security, respectively.

In algo trading, trading signals are generated by identifying the patterns for stocks (see for examples [2] and [3]) and for cryptocurrencies (see for example [4]) to maximize the returns/profits. Moreover, in genomics, with large genome sequencing projects becoming less expensive, pattern mining in such sequence data becomes an essential part of identifying risks for potential diseases to treat the individual accordingly. For capturing fraudulent activities, a dynamic pattern-mining procedure is required in online banking and network applications.

Financial firms need to discover hidden patterns in massive sets of data to keep track of the information in their data warehouse. Prediction of future financial events, such as stock markets, and foreign exchange rates, predictive financial and investment analysis, trading futures, and understanding and managing financial risks, are some of the key financial applications of data mining and machine learning. As a high volume of financial data becomes available, data-driven modelling has become the most attractive technology for various financial applications [5].

Financial analysis of data is very important in order to analyze whether the business is stable and profitable to make a capital investment. Financial analysts focus their analysis on the balance sheet, cash flow statement, and income statement.

Pattern mining techniques have been used to extract hidden patterns and predict future trends/forecasts and behaviours in financial markets. Advanced statistical, mathematical and artificial intelligence techniques are typically required for mining such data to generate trading signals and profits, especially high-frequency data such as cryptocurrencies. Moreover, pattern mining is a process of analyzing a large batch of information to discern trends and patterns. Pattern mining can also be used by corporations for everything from learning about what customers' buying preferences are, to fraud and anomaly detection, and spam filtering. Pattern mining programs break down patterns and connections in data [6] based on what information users request or provide.

In Thavaneswaran et al. [1] volatility parameter had been modelled as a fuzzy number and the superiority of the fuzzy forecasts over minimum mean square error forecasts were demonstrated. Existing Markov chain (MC) modelling of sequence data involve the use of crisp proportion parameters. However, in applications, the transition probabilities (TP) of the MC are estimated from the observed sequence data, and the corresponding standard error of the estimated TP can be used to model the transition probability matrix as a fuzzy matrix. In this paper instead of using standard errors of the TP estimates we propose a novel method to fuzzify the transition probability matrix.

In many practical applications, time series data are available as categorical data categorical time series data are scaled to

extract information. As the different scaling methods available are very subjective to user preference, the idea of optimal scaling (encoding) is introduced (see [7] for more details). This allows a researcher to extract maximum information from the data by implementing an optimal scaling approach. Further optimal scaling helps to obtain the spectral envelope of the data that reveals the cyclic patterns of the series.

Our objectives in this study include a) identifying desired nucleotide pattern in a DNA sequence and where it appears in the sequence; b) implementing computer security systems (a honeypot) for information systems by identifying the riskiest ports; c) avoiding potential risks of investments by watching for patterns in the log returns stocks prices and maximizing investment returns. We propose a novel pattern mining technique for this purpose. The main contribution of this paper is to fit an appropriate Markov chain model to a given data sequence and use the proposed fuzzy pattern mining algorithm to obtain resilient probabilistic forecasts and expected waiting time for patterns.

II. Background and Related Work

The Office of Science and Technology, USA, in 1989 had identified human genome sequencing as one of the high-performance computing grand challenge problems [8]. Since then a lot of research has been conducted to identify particular human genome sequences, and practising the genome sequencing knowledge for identifying diseases and remedying them has been happening with intelligent software systems [9]. A recent article reviews the state-of-the-art in science and technology of this challenging project that started 30 years ago [10]. A much recent article in Science [11] reported the completion of the remaining 8% of the human genome. Use of this knowledge has been commercialized for some time in the form of gene editing, and gene therapy kits. These days scientists work on genes that are mutated to control disease(es) in a person and hence improve their health by developing medicines to target those mutated genes. Pattern mining of DNA sequences is the foundation for all these activities.

Modern compute environments are mobile and dynamic, malicious attacks are constantly evolving (for example phishing attacks [12]) and the attack surface also are evolving with them. Therefore, mining available data for predicting potential attack scenarios is essential for the risk assessment of computer systems in any network. Binyamini et al. [13] presented an automated framework for modelling attack techniques from the text description of a security vulnerability. With such a framework, they could assess potential risks to the system. Several studies on countermeasures to such potential attacks were reported in the past [14] and they are also subject to risk avoidance.

In finance, one traditional way to decide on buying or sell of assets is by mean reversion process, which is a theory that an asset's price will tend to converge to the average price over time [15]. In other words, deviations from the average price are expected to revert to the average. This knowledge serves as the cornerstone of multiple trading strategies in the financial market. There are multiple technical analyses in making buy or sell decisions - see for example forecasting directional change strategy [16]. Mining a pattern of price movement would serve as a potential strategy to buy/sell/hold, the focus of our study in this application domain.

III. Approach and Algorithms

A. Traditional Approach (Matrix Approach)

The first pattern mining approach we discuss in this study is the traditional approach. The traditional approach requires constructing a transitional probability matrix and finding the inverse of a fundamental matrix (see [17] for more details). Thus, for convenience, we refer traditional approach as the matrix approach in this study. The matrix approach calculates the expected waiting time of a desired pattern first appearing in a given sequence. The basic steps of the matrix approach are summarized in Algorithm 1, and we discuss the expected waiting times of three applications of pattern mining in the experimental section.

Algorithm 1 Traditional Approach (Matrix Approach)

Require: Finite state space and corresponding transition probabilities from one state to another

1: Construct a transition probability matrix ($\tilde{P}$) for the given state space.
2: Identify the desired pattern sequence of the states.
3: Consider a sequential search from the first state of the desired pattern until we reach the last state of the desired pattern.
4: Following the sequential search, construct a pattern matrix (transition probability matrix of the desired pattern) (P) with the desired pattern as an absorbing state.
5: Calculate the fundamental matrix (F) by inverting the matrix $(I - Q)$ where I is a identity matrix and Q is a minor matrix obtained from P after deleting last row and last column.
6: Calculate the Expected waiting time (expected number of data points until we reach the desired pattern for the first time) by adding first-row elements of F.

The matrix approach required a transition matrix with probability estimates for the jumps from one state to another. Corresponding probabilities are estimated, and there is always uncertainty associated with the estimates. However, the matrix approach can not incorporate the uncertainty of probability estimates in the expected waiting time computation. Furthermore, the method requires computing the inverse of a matrix to find the expected waiting time. Thus, one can expect limitations to implement the matrix approach in practice as there is always uncertainty associated with estimated probabilities and finding the inverse of the matrix may not be possible. Thus, to overcome the limitations of the matrix approach, we propose a novel data-driven fuzzy pattern mining algorithm in this study. We also extend our data-driven fuzzy pattern mining algorithm for both categorical sequences (e.g., DNA

sequences) and numerical sequences (e.g., asset prices), and they are summarized in Algorithm 2 and Algorithm 3.

Algorithm 2 Data-Driven Fuzzy Pattern Mining Algorithm for Categorical Sequences

Require: Data: Categorical sequence data with finite state space. Parameters: Let k be the number of distinct states.

1: Identify the distinct element of the state space.
2: Use R function `markovchainFit` from the R package `Markovchain` [18] to estimate the transition probability matrix. ($\tilde{P}$).
3: Construct a stochastic matrix (A) where A is $k \times k$ matrix with $a_{ij} = 1/k$ for each $i, j = 1, \ldots, k$.
4: Calculate the alpha cuts of $\tilde{P}$. $P(\alpha) = \alpha * A + (1-\alpha) * \tilde{P}$ where $\alpha \in [0, 1]$ $\{p_{ij}(\alpha) = \alpha * a_{ij} + (1-\alpha) * \tilde{p}_{ij}\}$.
5: Simulate distinct N number of sequences with given states until the desired pattern first appears. Record the number of steps ($n_l \,; l = 1, \ldots, N$) until the desired pattern appears first for each of the sequences.
6: Calculate the expected waiting time by taking the average number of steps to reach the desired pattern ($\sum_{l=1}^{N} n_l/N$).

Algorithm 3 Data-Driven Fuzzy Pattern Mining Algorithm for Numerical Sequences

Require: Data: Adjusted closing price of stocks/indexes and Cryptocurrencies ($S_t, t = 0, \ldots, n$). Parameters: Let k be the number of distinct states.

1: $r_t \leftarrow \log S_t - \log S_{t-1} \,; t = 1, \ldots, n$
2: signals $\leftarrow$ 1 if $r_t \geq 0$ o.w. signals $\leftarrow$ -1 {A dataframe called signals is created according to the symbol of r_t values}
3: Use R function `markovchainFit` from the R package `Markovchain` [18] to estimate the transition probability matrix. ($\tilde{P}$)
4: Construct a stochastic matrix (A) where A is $k \times k$ matrix with $a_{ij} = 1/k$ for each $i, j = 1, \ldots, k$.
5: Calculate the alpha cuts of $\tilde{P}$. $P(\alpha) = \alpha * A + (1-\alpha) * \tilde{P}$ where $\alpha \in [0, 1]$ $\{p_{ij}(\alpha) = \alpha * a_{ij} + (1-\alpha) * \tilde{p}_{ij}\}$
6: Simulate distinct N number of sequences with given states until the desired pattern first appears. Record the number of steps ($n_l \,; l = 1, \ldots, N$) until the desired pattern appears first for each of the sequences.
7: Calculate the expected waiting time by taking the average number of steps to reach the desired pattern ($\sum_{l=1}^{N} n_l/N$).

IV. Experimentation and Real Data Analysis

A. Application I: Successive Occurrences of DNA Nucleotides

Bioinformatics is the science of managing, mining, and interpreting information from biological data (DNA or protein sequences), and pattern mining plays a key role in capturing fundamental problems in this field.

The nucleotides are the building blocks of nucleic acids, and nucleic acids are most commonly known as RNA and DNA. These essential biomolecules can be found in all life forms on Earth. Mainly there are two ways of absorbing nucleotides: 1) diet and 2) synthesized in the liver from common nutrients. Nucleotides contain instructions for building an organism. Thus, extending the understanding of nucleotide sequences improves the knowledge of genetic functions. The genetic code, also called DNA, can be represented by four letters (A, C, G, and T), and they represent four types of nucleotides based on which nitrogenous bases they contain [19]. A certain sequence of the nucleotides reveals how the other substances in the cells are controlled and the possibility of initiating cancers in some forms. For illustration purposes, here in this study, we discuss the expected waiting time or the average number of nucleotides needed to reach the pattern ACCGC first time using the matrix approach and novel pattern mining algorithm with fuzzy matrices using Algorithms 1 and 2.

The R package `astsa` [20] contains several datasets covering different fields of study. This study investigates the dataset *bnrf1ebv* (Nucleotide sequence - BNRF1 Epstein-Barr), which contains the nucleotide sequence of the BNRF1 gene of the Epstein-Barr virus (EBV) with 3954 bp (base pairs). In *bnrf1ebv*, the format of the data is "AGAATTCGTCTT..." and for convenience data are represented as "131144234244..." where $1 = A, 2 = C, 3 = G$, and $4 = T$. Figure 1 summarizes the base proportions of nucleotides of the BNRF1 gene of the Epstein-Barr virus. Note that in the Epstein-Barr virus gene sequence, the base proportions of nucleotides C and G are high compared to the base proportions of nucleotides A and T.

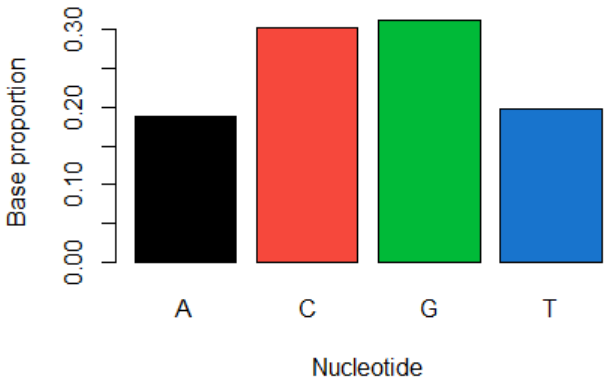

Fig. 1. Base proportions of nucleotide in sequence data *bnrf1ebv*

A compound containing two nucleotides is known as a dinucleotide. Studying further dinucleotides is important as they reveal more information about different functions, and for example, consider the functions of NAD+ (nicotinamide adenine dinucleotide). It is the most common and abundant molecule in single-cell organisms (e.g., bacteria) to multicellular organisms (e.g., primates). NAD+ helps convert food to energy and ensures cell functions are properly executed. If the cell functions are not properly executed, the bodies will be ageing rapidly and exposed to diseases [21]. For the nucleotide

sequence of the BNRF1 gene of the Epstein-Barr virus, base proportions of dinucleotide are shown in Figure 2. Note that, the observed proportion for AA ($P(AA)$) is approximately 0.033 and it is very close to two times the observed proportion of A ($P(A) * P(A) = 0.188 * 0.188$).

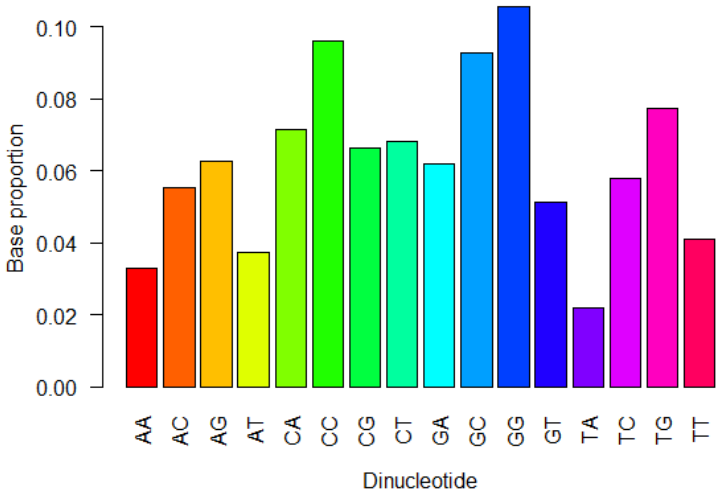

Fig. 2. Base proportions of dinucleotide in sequence data *bnrf1ebv*

Often, the concept of scaling is implemented to extract relevant information for categorical time series. Stofer et al. [22] introduced the idea of optimal scaling and spectral envelope leading to efficient estimates , and the goal is to find the scaling method which brings all the interesting features in the data. The spectral envelope conveniently envelopes the standardized spectrum of any scaled process. Most of the work on DNA sequence analysis involves pattern alignment and spectral envelope efficiently discovering the periodic components in the series. The spectral envelope for the entire coding sequence of *bnrf1ebv* is given in Figure 3 and observes the strong signal at frequency $1/3$ (the periodic notion of the signals).

Dynamic spectral envelope estimates with a block size of 500 for the BNRF1 gene (bp 1736-5689) of the Epstein–Barr virus (EBV) are given in Figure 4. The vertical dashed lines indicate the blocks, and darker regions indicate values over the approximate 0.005 null significance threshold. Observe that except for the end of the sequence, a basic cyclic pattern exists through the gene (darker lines at frequency $1/3$). Thus, this reveals some pattern anomalies in the latter part of the sequence.

Following the steps in Algorithm 2 and considering successive occurrences of DNA nucleotides for *bnrf1ebv*, a transition

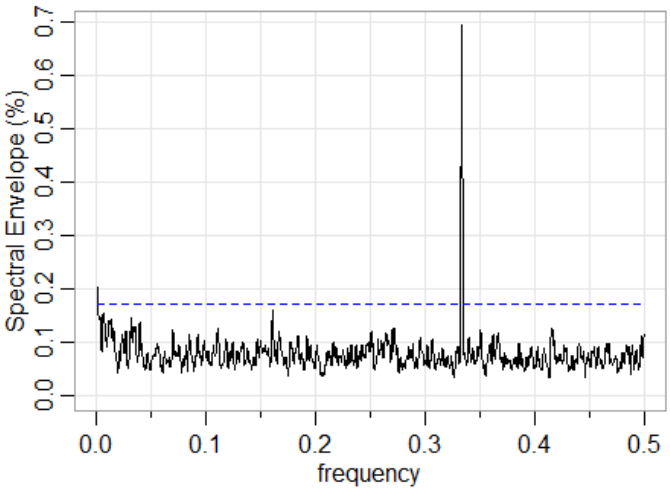

Fig. 3. Spectral Envelop for sequence data *bnrf1ebv*

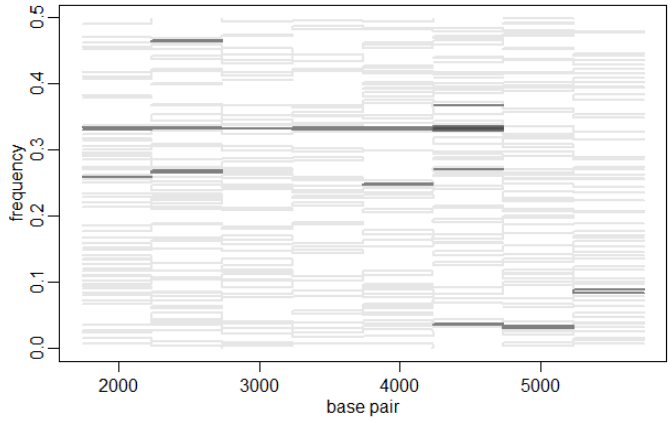

Fig. 4. Dynamic Spectral Envelop for sequence data *bnrf1ebv*

probability matrix ($\tilde{P}$) is obtained:

$$\tilde{P} = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{array}{c} \begin{array}{cccc} A & C & G & T \end{array} \\ \begin{pmatrix} 0.175 & 0.294 & 0.332 & 0.199 \\ 0.237 & 0.318 & 0.219 & 0.226 \\ 0.198 & 0.298 & 0.339 & 0.165 \\ 0.110 & 0.292 & 0.391 & 0.207 \end{pmatrix} \end{array}$$

Following the transition matrix, an absorbing Markov chain is constructed for the patterns A, AC, ACC, ACCG, and ACCGC. Note that starting state for the absorbing Markov chain is the null set.

$$\begin{array}{c} \\ \phi \\ A \\ AC \\ ACC \\ ACCG \\ ACCGC \end{array} \begin{array}{c} \begin{array}{cccccc} \phi & A & AC & ACC & ACCG & ACCGC \end{array} \\ \begin{pmatrix} 0.825 & 0.175 & 0 & 0 & 0 & 0 \\ 0.531 & 0.175 & 0.294 & 0 & 0 & 0 \\ 0.445 & 0.237 & 0 & 0.318 & 0 & 0 \\ 0.544 & 0.237 & 0 & 0 & 0.219 & 0 \\ 0.504 & 0.198 & 0 & 0 & 0 & 0.298 \\ 0 & 0 & 0 & 0 & 0 & 1.000 \end{pmatrix} \end{array}$$

Observe that when the chain moves to state ACCG from ACC, the corresponding transition probability is 0.219. This probability coincides with the probability in the transition matrix for moving from C to G. Note that each jump corresponds to a one-step jump from one state to another. Thus, it can move from state C to state A (second entry in the fourth row), and it is not possible to move from state C to state AC. The first column will be completed with the complementary probability, making the absorbing matrix a Markov chain.

The R function `markovchainFit` also produces confidence matrices of estimated transition probabilities. For 95% confidence level `lowerEndpointMatrix` and `upperEndpointMatrix` are given below. Observe that for both lower and upper endpoint matrices, row sums do not add to 1. Thus, both lower and upper endpoint matrices are not Markov chains. Therefore, even though lower and upper matrices help to capture the uncertainty of the estimates, they are not useful to construct α-cuts of the probability estimates. In contrast, fuzzy matrices proposed in this paper ($P(\alpha) = \alpha A + (1-\alpha)\tilde{P}$) always produce Markov chains and can be used to construct α-cuts for a given transition matrix.

Lower Endpoint Matrix of $\tilde{P}$:

	A	C	G	T
A	0.145	0.255	0.291	0.167
C	0.209	0.286	0.193	0.199
G	0.173	0.268	0.306	0.142
T	0.087	0.255	0.347	0.175

Upper Endpoint Matrix of $\tilde{P}$:

	A	C	G	T
A	0.205	0.333	0.373	0.231
C	0.264	0.350	0.246	0.253
G	0.223	0.329	0.371	0.188
T	0.133	0.330	0.435	0.239

Following paragraphs discussed findings from both the matrix approach and novel fuzzy pattern mining algorithm for *bnrf1ebv*. A fundamental matrix F is calculated following the steps of Algorithm 1, and the average number of nucleotides needed to reach the desired pattern ACCGC for the first time is 913.60. The proposed pattern mining algorithm with fuzzy matrices suggests that the expected waiting time of 913.20 when α equals 0.288 (see Algorithm 2). Note that the expected waiting times are close, and α is not equal to zero. Thus, incorporating stochastic variation in the transition probability estimates through fuzzy matrices, the new approach provides an alternative path to produce α-cuts for transition matrices. Furthermore, first-row and third-row entries of $\tilde{P}$ are very close indicating they are linearly related. This would lead to a very small value of determinant and ultimately inverse of $\tilde{P}$ may be undefined. Table I summarizes observed and estimated counts from stationary probabilities of nucleotides from the sequence itself and Markov chain, respectively . Note that counts are very close suggesting that for a long sequence, the Markov chain converges.

TABLE I
OBSERVED AND ESTIMATED COUNTS OF NUCLEOTIDES FOR *bnrf1ebv*

	Nucleotide			
	A	C	G	T
Observed Counts	744	1195	1232	783
Estimated Counts	743.214	1195.306	1232.316	783.164

B. Application II: Computer Security Applications

In computer security, a mechanism designed to identify, divert or even counterattack attempts of unauthorized access to information systems is known as a honeypot [23] . Intruders trying to gain access to protected systems are a new threat that needs innovative solutions in the information era. A study conducted by Kimou et al. in 2010 [24] observes data from the www.Leurre.com project that contains information about the number of observed attacks on different ports (e.g. 80 (HTTP - Hypertext Transfer Protocol), 135 (EPMAP - End Point Mapper), 139 (NETBIOS-SSN - Network Basic Input/Output System-Session Service), and 445 (MicroSoft-DS - Microsoft Directory Services)). These authors have considered the most attacked ports in their study to estimate the Markov transition matrix for weekly attacks:

	80	135	139	445	No Attack
80	0	0	0	0	1
135	0	8/13	3/13	1/13	1/13
139	1/16	3/16	3/8	1/4	1/8
445	0	1/11	4/11	5/11	1/11
NO Attack	0	1/8	1/2	1/8	1/4

Once the transition probabilities are estimated several thrilling questions arise. One may ask what is the expected waiting time for observing the event "No Attacks" for four consecutive weeks (for a month), and what port is most likely to get attacked after four weeks (i.e. probability of a port getting attacked after four weeks) when the initial state is "No Attack". In other words, we are interested in unique patterns of transition matrix labels and their corresponding probabilities.

First, the matrix approach is used and the expected waiting time is 340 weeks until we observe four consecutive "No Attacks" (i.e. no attacks for a month) (Algorithm 1). On the other hand, the proposed novel pattern mining algorithm with fuzzy matrices (Algorithm 2), which avoids inverting larger order matrices, provides the expected waiting time for observing " No Attacks for a month" as 338.4 weeks with the optimal value of α as 0.391. The Markov chain below provides the four weeks ahead resilient (more robust) probabilistic forecasts (with α equals 0.391) as:

	80	135	139	445	No Attack
80	0.089	0.217	0.282	0.204	0.208
135	0.089	0.226	0.279	0.201	0.206
139	0.089	0.219	0.282	0.204	0.207
445	0.089	0.217	0.282	0.205	0.207
NO Attack	0.089	0.218	0.282	0.204	0.207

Observe that, starting from the state "No Attack", port 80 (HTTP) is least likely to get attacked and port 139 (NETBIOS-

SSN) is most likely to get attacked in 4 weeks with respective probabilities of 0.089 and 0.218. For both cases α equals 0.391. Note that each entry in matrix A is the same $(1/k)$, and it is corresponding to the transition probability when we do not hold any information about the likelihood of going from one state to another. Thus, with matrix A, we believe each transition has an equal chance. Matrix P is fixed, and we believe we have the perfect information (we know exact transition probabilities). Fuzzyfying the matrix $\tilde{P}$ allow us to account for stochastic variation in the transition probability estimates and provide fuzzy estimates of transition probability and expected waiting time.

C. Application III: Financial Applications

Stock is a security which represents the ownership of a fraction of a corporation. Corporation offers stocks to raise funds for their corporation activities. Stocks of blue chip companies traded at high prices in stock markets as stakeholders believe their values would further increase in the future. However, due to market fluctuation and different corporate actions, stock value is expected to vary every second. Thus, observing and predicting the general direction of the price change is important for a good investment. In this study, we consider the daily adjusted closing price of stocks (SP500-Standard and Poor's 500, VIX-CBOE Volatility Index, AAPL-Apple Inc., and GOOG-Alphabet Inc.) in the numerical analysis. Adjusted closing price provides an accurate snapshot of the stock value after adjusting for various factors such as dividend payouts. Furthermore, we extend our numerical study by considering six cryptocurrencies (BTC-Bitcoin, ETH-Ethereum, BNB-Binance Coin, XRP-Ripple, DOGE-Dogecoin, and ADA-Cardano) based on their popularity and market capital. Cryptocurrencies are decentralized digital currencies and their value is purely driven by the trust placed in them (see [26] for more details). Figures 5 and 6 represent adjusted closing price changes from January 2017 to November 2022 and from November 2021 to November 2022 for stocks and cryptocurrencies, respectively. Observe the price increase decreases over the study period. Profitable stocks/cryptocurrencies in one period may lead to a loss in investment in another period due to price drops. Thus, this emphasises the importance of efficient pattern prediction techniques that could lead to profitable investments by buying/selling stocks/cryptocurrencies at the right time.

In the study, we observe a pattern of the length of four (four days) to estimate the expected waiting time. We generate a binary signal by observing the sign of the log return of adjusted closing prices (see Algorithm 3). If the log return is positive, the generated signal is 1, and if the log return is negative, the generated signal is 0. We have considered daily adjusted closing prices from 2021-11-26 to 2022-11-26 for cryptocurrencies and from 2017-01-01 to 2022-11-30 for stocks. It is important to note that the adjusted closing prices are available throughout the week for cryptocurrencies. However, adjusted closing prices are only available on weekdays for stocks as the stock market only operates during the weekdays.

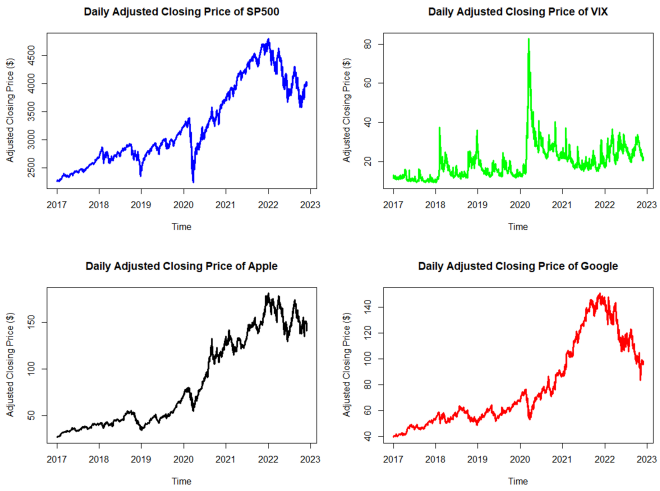

Fig. 5. Adjusted Closing Prices of Stocks (Jan 2017 - Nov 2022)

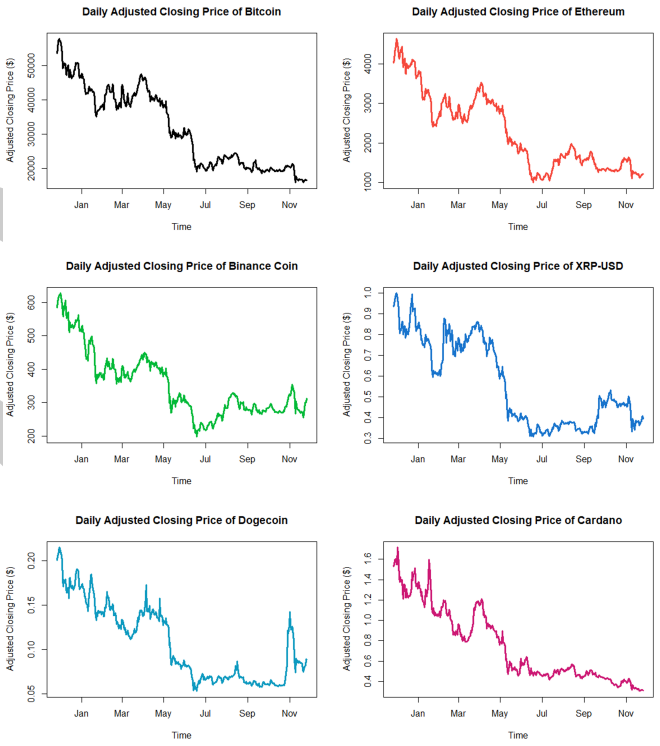

Fig. 6. Adjusted Closing Prices of Cryptocurrencies (Nov 2021 - Nov 2022)

A transition matrix $(\tilde{P})$ of two states (state 1 $(r_t \geq 0)$ and state -1 $(r_t < 0)$ can be defined by reporting only the diagonal elements (p_{00}, p_{11}). Off-diagonal elements (p_{01}, p_{10}) can be calculated by considering the complement of diagonal transition probabilities $(p_{01} = 1 - p_{00}, p_{10} = 1 - p_{11})$. Hence we can report complete transition probabilities for all the states (signals) of stocks/indexes and cryptocurrencies by reporting only the diagonal elements. Diagonal elements for the selected stocks/indexes and cryptocurrencies are; SP500 $(0.432, 0.529)$,

VIX (0.537, 0.433), GOOG (0.457, 0.546), AAPL (0.448, 0.521), BTC (0.536, 0.480), ETH (0.526, 0.468), BNB (0.473, 0.434), XRP (0.454, 0.439), DOGE (0.455, 0.420) and ADA (0.474, 0.409).

Table II and III summarize the expected waiting time (in days) using the novel algorithm for each of the patterns of length four for stocks/indexes (SP500, VIX and GOOG) and cryptocurrencies (BTC, ETH, and BNB). They also report optimal α values that give an expected waiting time closer to the matrix approach for each pattern. It can be seen that α changes with the patterns. This illustrates the importance of the new fuzzy approach to model risk of probability estimates. Among the stocks/indexes, both SP500 and Google have the highest expected waiting time to reach the pattern "0000", and for VIX, the highest expected waiting time is for the pattern "1111". Thus, one can expect a price drop for four consecutive days shortly for VIX and a price increase for four consecutive days in one and a half months for both SP500 and Google. Similar to VIX, for BTC, ETH and BNB, the highest expected time is for pattern "1111". Therefore, a consecutive price increase for four days takes the longest time for BTC, ETH, and BNB.

TABLE II
FUZZY EXPECTED WAITING TIME WITH α FOR SP500, VIX, AND GOOGLE

	SP500		VIX		Google	
Pattern	Waiting Time	α	Waiting Time	α	Waiting Time	α
0000	44.54	0.10	23.65	0.20	39.37	0.14
0001	19.47	0.10	14.65	0.19	19.16	0.21
0010	22.54	0.10	15.58	0.17	21.29	0.21
0100	20.46	0.10	15.79	0.13	21.68	0.15
1000	20.12	0.10	13.70	0.16	19.29	0.20
0011	17.10	0.10	17.93	0.16	16.77	0.23
0101	18.99	0.20	19.12	0.10	20.30	0.60
1001	17.57	0.90	17.93	0.30	18.13	0.24
0110	18.02	0.50	17.89	0.17	18.49	0.17
1010	18.55	0.20	18.88	0.17	20.41	0.18
1100	16.85	0.10	17.27	0.28	17.34	0.25
0111	14.26	0.20	20.80	0.12	13.59	0.22
1110	16.46	0.70	20.80	0.10	14.70	0.24
1101	17.96	0.90	21.83	0.10	16.65	0.25
1011	17.81	0.80	22.11	0.13	16.72	0.26
1111	24.79	0.10	46.31	0.14	22.74	0.12

Table IV shows the expected waiting time (in days) to observe the pattern "0101" for stocks/indexes and cryptocurrencies under both approaches.

V. CONCLUSION

For many Markov chains that arise in applications (network security, health, finance, etc.), state spaces are huge, and existing matrix methods may not be practical or even not possible to implement. This paper first presents a novel fuzzy transition probability matrix and a novel pattern mining algorithm. The proposed algorithm, which avoids the inversion of the pattern matrix is applicable to Markov chains in a wider context, with huge state spaces. The driving idea, unlike the existing work, is that the resilient pattern mining algorithm

TABLE III
FUZZY EXPECTED PATTERN WAITING TIME FOR BTC, ETH AND BNB

	BTC		ETH		BNB	
Pattern	Waiting Time	α	Waiting Time	α	Waiting Time	α
0000	24.56	0.10	25.43	0.29	34.13	0.12
0001	15.45	0.30	15.49	0.19	17.02	0.21
0010	17.94	0.29	17.04	0.20	16.80	0.28
0100	17.76	0.19	17.39	0.30	16.73	0.25
1000	14.21	0.24	14.45	0.11	16.33	0.16
0011	16.27	0.18	16.70	0.14	18.15	0.12
0101	20.81	0.15	19.78	0.40	17.23	0.30
1001	18.54	0.21	18.09	0.25	16.59	0.27
0110	18.21	0.16	17.83	0.12	17.35	0.17
1010	20.93	0.19	19.83	0.28	15.98	0.20
1100	16.72	0.27	16.99	0.16	18.04	0.18
0111	17.61	0.25	18.20	0.15	19.47	0.21
1110	17.90	0.17	18.48	0.14	19.73	0.14
1101	20.56	0.13	20.74	0.14	19.08	0.13
1011	21.16	0.11	20.72	0.15	18.90	0.22
1111	34.96	0.17	37.16	0.25	43.57	0.16

TABLE IV
EXPECTED WAITING TIME FOR THE PATTERN 0101 USING MATRIX AND FUZZY APPROACHES

		α	Waiting Time*	Waiting Time**
Stock/Index	SP500	0.20	18.97	18.99
	VIX	0.10	19.13	19.12
	GOOG	0.60	20.40	20.30
	AAPL	0.30	19.07	19.07
Cryptocurrency	BTC	0.15	20.81	20.81
	ETH	0.40	19.87	19.78
	BNB	0.30	16.80	17.23
	XRP	0.20	16.44	16.15
	DOGE	0.30	15.92	15.99
	ADA	0.20	16.12	16.05

Waiting Time* - Expected waiting time using matrix approach
Waiting Time** - Expected waiting time using fuzzy approach

is obtained by fuzzifying the transition probability matrix. Three applications (DNA sequence data, patterns generated by the log returns of the cryptocurrencies and Markov chain model for network security) of the proposed algorithm are discussed in detail. The main contribution of this paper is to fit an appropriate Markov chain model to a given data sequence and use the proposed fuzzy pattern mining algorithm to obtain resilient probabilistic forecasts and expected waiting time for patterns. Often, the concept of scaling is implemented to extract relevant information for categorical time series. Optimal scaling and spectral envelope leading to efficient estimates are also discussed in some detail.

VI. APPENDIX

Now we provide a theorem on the expected waiting time for Markov chain generated patterns. Note that here the length of the desired pattern is four and diagonal elements and off-diagonal elements of the transition matrix are equal.

Transition probability matrix:

$$\tilde{P} = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \begin{pmatrix} \mathrm{p} & \mathrm{q} \\ \mathrm{q} & \mathrm{p} \end{pmatrix} \end{array}$$

Theorem; (a) The expected waiting time for patterns 1111 and 0000 is given by

$$1 + \frac{2}{qp^3} - \frac{3}{2q},$$

(b) The expected waiting time for the patterns 0110 and 1001 is given by

$$1 + \frac{1}{2q} + \frac{2}{pq^2},$$

(c) The expected waiting time for patterns 0011 and 1100 is given by

$$1 + \frac{2}{qp^2} - \frac{1}{2q},$$

(d) The expected waiting time for the patterns 0101 and 1010 is given by

$$1 + \frac{2}{q^3} + \frac{3}{2q^3}.$$

Let $p = 1/4$ and $q = 3/4$. For 10,000 simulated sequences, expected waiting times are summarized in Table V.

TABLE V
EXPECTED WAITING TIME FOR THE PATTERN WITH LENGTH FOUR

Pattern	Expected Waiting Time
0000	169.401
1000	42.7225
0100	16.0454
0010	15.8352
0001	44.0708
1100	42.8327
0110	15.9563
0011	42.7726
1010	7.7926
1001	16.1512
0101	7.7593
1110	42.8367
0111	42.9294
1011	15.8434
1101	15.9493
1111	171.846

The expected waiting time for transition probabilities for a transition matrix with two states (0 and 1) is provided here. This can be easily extended for more than two states and desired patterns with lengths greater than four. For 10,000 simulated sequences, expected waiting times are summarized in Table VI for a pattern with lengths of five and two states. Pseudocode to calculate expected waiting time:

```
p = 1/4
q = 1-p
Ptilde <- matrix(c(p,q,q,p), nrow = 2, ncol = 2)
init <- c(1/2,1/2) # Initial Distribution
# Function - Expected Waiting Time
EWTsimulation = function(Ptilde,pattern,init){
  sequences = 10000
  states <- 1:length(init)
  simlistTEMP <- 0
  for (j in 1:sequences) {
    simlist <- c()
    simlist[1] <- sample(states,1,prob=init)
    simlist[2] <- sample(states,1,prob=Ptilde[
    simlist[1],])
    simlist[3] <- sample(states,1,prob=Ptilde[
    simlist[2],])
    simlist[4] <- sample(states,1,prob=Ptilde[
    simlist[3],])
    .
    .
    .
    simlist[length(pattern)] <- sample(states,1,prob
    =Ptilde[simlist[length(pattern)-1],])
    STATE <- c(simlist[1], simlist[2], simlist[3],
    simlist[4],...,simlist[length(pattern)])
    i <- length(pattern)
    k <- length(pattern)
    while (!prod(labels[STATE]==pattern))
    {
      simlist[i+1] <- sample(states,1,prob=Ptilde[
    simlist[i],])
      STATE <- c(simlist[i-(length(pattern)-2)], ...
     , simlist[i-2], simlist[i-1], simlist[i],
    simlist[i+1])
      k <- k + 1
      i <- i + 1
    }
    simlistTEMP[j] <- k
  }
  return(mean(simlistTEMP))
}
```

TABLE VI
EXPECTED WAITING TIME FOR THE PATTERN WITH LENGTH FIVE

Pattern	Expected Waiting Time	Pattern	Expected Waiting Time
00000	673.677	00001	169.282
10000	170.778	10001	58.314
01000	59.358	01001	22.078
00100	65.797	00101	19.294
00010	58.705	00011	170.350
11000	167.999	11001	58.185
01100	58.011	01101	22.060
00110	59.294	00111	58.121
10100	19.229	10101	19.093
10010	21.838	10011	57.689
01010	11.486	01011	19.383
11100	172.831	11101	58.617
01110	58.156	01111	172.850
10110	22.161	10111	59.364
11010	19.499	11011	65.929
11110	167.716	11111	672.075

REFERENCES

[1] Thavaneswaran, Aerambamoorthy, Srimantoorao S. Appadoo, and Alex Paseka. "Weighted possibilistic moments of fuzzy numbers with applications to GARCH modeling and option pricing." Mathematical and Computer Modelling 49, no. 1-2, pp. 352-368, 2009.

[2] Liang, You, Aerambamoorthy Thavaneswaran, Na Yu, Md Erfanul Hoque, and Ruppa K. Thulasiram. "Dynamic data science applications in optimal profit algorithmic trading." In 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1314-1319. IEEE, 2020.

[3] Thavaneswaran, A., You Liang, Zimo Zhu, and Ruppa K. Thulasiram. "Novel data-driven fuzzy algorithmic volatility forecasting models with applications to algorithmic trading." In 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8. IEEE, 2020.

[4] Thavaneswaran, Aerambamoorthy, You Liang, Sulalitha Bowala, Alex Paseka, and Melody Ghahramani. "Deep Learning Predictions for Cryptocurrencies." In 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1280-1285. IEEE, 2022.
[5] Singh, Japjeet, Sulalitha Bowala, Aerambamoorthy Thavaneswaran, Ruppa Thulasiram, and Saumen Mandal. "Data-Driven and Neuro-Volatility Fuzzy Forecasts for Cryptocurrencies." Proc. of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1-8, 2022.
[6] Friedman, Jerome H. "Data Mining and Statistics: What's the connection?." Computing science and statistics 29, no. 1, pp. 3-9, 1998.
[7] Davis, Richard A., Scott H. Holan, Robert Lund, and Nalini Ravishanker, eds. Handbook of discrete-valued time series. CRC Press, 2016.
[8] United States. Office of Science, and Technology Policy. The federal high performance computing program. Executive Office of the President, Office of Science and Technology Policy, 1989.
[9] Hood, Leroy, and Lee Rowen. "The human genome project: big science transforms biology and medicine." Genome medicine 5, no. 9, pp. 1-8, 2013.
[10] Gibbs, Richard A. "The human genome project changed everything." Nature Reviews Genetics 21, no. 10,pp 575-576, 2020.
[11] Nurk, Sergey, Sergey Koren, Arang Rhie, Mikko Rautiainen, Andrey V. Bzikadze, Alla Mikheenko, Mitchell R. Vollger et al. "The complete sequence of a human genome." Science 376, no. 6588, pp. 44-53, 2022.
[12] Sun, Yuanyi, Sencun Zhu, Yan Zhao, and Pengfei Sun. "A User-Friendly Two-Factor Authentication Method against Real-Time Phishing Attacks." In 2022 IEEE Conference on Communications and Network Security (CNS), pp. 91-99. IEEE, 2022.
[13] Binyamini, Hodaya, Ron Bitton, Masaki Inokuchi, Tomohiko Yagyu, Yuval Elovici, and Asaf Shabtai. "A Framework for Modeling Cyber Attack Techniques from Security Vulnerability Descriptions." In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2574-2583. 2021.
[14] Xinming, Ou, S. Govindavajhala, and A. W. Appel. "A logic-based network security analyzer." In 14th USENIX Security Symposium; http://www. usenix. org/events/sec05/tech/ou. html: Baltimore, Maryland, USA. 2005.
[15] Damghani, Babak Mahdavi. "The non-misleading value of inferred correlation: An introduction to the Cointelation Model." Wilmott 2013, no. 67, pp. 50-61, 2013.
[16] Bakhach, Amer M., Edward PK Tsang, and V. L. Raju Chinthalapati. "TSFDC: A trading strategy based on forecasting directional change." Intelligent Systems in Accounting, Finance and Management 25, no. 3, 105-123, 2018.
[17] Dobrow, Robert P. Introduction to stochastic processes with R. John Wiley & Sons, 2016.
[18] Spedicato, Giorgio Alfredo, Tae Seung Kang, Sai Bhargav Yalamanchi, Deepak Yadav, and Ignacio Cordón. "The markovchain package: a package for easily handling Discrete Markov Chains in R." Accessed Dec (2016).
[19] Gentleman, Jane F., and Ronald C. Mullin. "The distribution of the frequency of occurrence of nucleotide subsequences, based on their overlap capability." Biometrics, pp. 35–52, 1989.
[20] Stoffer, David, and Maintainer David Stoffer. "Package 'astsa'." blood 8 (2022): 1.
[21] Ying, Weihai. "NAD+ and NADH in cellular functions and cell death." Frontiers in Bioscience-Landmark 11, no. 3 (2006): 3129-3148.
[22] Stoffer, David S., David E. Tyler, and Andrew J. McDougall. "Spectral analysis for categorical time series: Scaling and the spectral envelope." Biometrika 80, no. 3 (1993): 611-622.
[23] Spitzner, Lance. Honeypots: tracking hackers. Vol. 1. Reading: Addison-Wesley, 2003.
[24] Kimou, K. P., B. Barry, M. Babri, S. Oumtanaga, and T. L. Kadjo. "An efficient analysis of honeypot data based on Markov chain." Journal of Applied Sciences 10, no. 3 (2010): 196-202.
[25] Shumway, Robert H., David S. Stoffer, and David S. Stoffer. Time series analysis and its applications. Vol. 3. New York: springer, 2000.
[26] Bowala, Sulalitha, Japjeet Singh, Aerambamoorthy Thavaneswaran, Ruppa Thulasiram, and Saumen Mandal. "Comparison of Fuzzy Risk Forecast Intervals for Cryptocurrencies." In 2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr), pp. 1-8. IEEE, 2022.