Old Dominion University
ODU Digital Commons

Mathematics & Statistics Faculty Publications | Mathematics & Statistics

4-2022

A Super Fast Algorithm for Estimating Sample Entropy

Weifeng Liu
Sun Yat-sen University

Ying Jiang
Sun Yat-sen University

Yuesheng Xu
Old Dominion University, y1xu@odu.edu

Follow this and additional works at: https://digitalcommons.odu.edu/mathstat_fac_pubs

 Part of the Mathematics Commons, Physics Commons, and the Theory and Algorithms Commons

Original Publication Citation

Liu, W., Jiang, Y., & Xu, Y. (2022). A super fast algorithm for estimating sample entropy. *Entropy, 24*(4), Article 524. https://www.mdpi.com/1099-4300/24/4/524

This Article is brought to you for free and open access by the Mathematics & Statistics at ODU Digital Commons. It has been accepted for inclusion in Mathematics & Statistics Faculty Publications by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.

entropy

MDPI

Article

A Super Fast Algorithm for Estimating Sample Entropy

Weifeng Liu [1], Ying Jiang [1,*] and Yuesheng Xu [2]

[1] Guangdong Province Key Laboratory of Computational Science, School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China; liuwf27@mail2.sysu.edu.cn
[2] Department of Mathematics and Statistics, Old Dominion University, Norfolk, VA 23529, USA; y1xu@odu.edu
* Correspondence: jiangy32@mail.sysu.edu.cn

Abstract: Sample entropy, an approximation of the Kolmogorov entropy, was proposed to characterize complexity of a time series, which is essentially defined as $-\log(B/A)$, where B denotes the number of matched template pairs with length m and A denotes the number of matched template pairs with $m+1$, for a predetermined positive integer m. It has been widely used to analyze physiological signals. As computing sample entropy is time consuming, the box-assisted, bucket-assisted, x-sort, assisted sliding box, and kd-tree-based algorithms were proposed to accelerate its computation. These algorithms require $O(N^2)$ or $O(N^{2-\frac{1}{m+1}})$ computational complexity, where N is the length of the time series analyzed. When N is big, the computational costs of these algorithms are large. We propose a super fast algorithm to estimate sample entropy based on Monte Carlo, with computational costs independent of N (the length of the time series) and the estimation converging to the exact sample entropy as the number of repeating experiments becomes large. The convergence rate of the algorithm is also established. Numerical experiments are performed for electrocardiogram time series, electroencephalogram time series, cardiac inter-beat time series, mechanical vibration signals (MVS), meteorological data (MD), and $1/f$ noise. Numerical results show that the proposed algorithm can gain 100–1000 times speedup compared to the kd-tree and assisted sliding box algorithms while providing satisfactory approximate accuracy.

Keywords: entropy; sample entropy; fast algorithm; Monte Carlo method

check for updates

Citation: Liu, W.; Jiang, Y.; Xu, Y. A Super Fast Algorithm for Estimating Sample Entropy. *Entropy* **2022**, *24*, 524. https://doi.org/10.3390/e24040524

Academic Editor: Karsten Keller

Received: 27 February 2022
Accepted: 2 April 2022
Published: 8 April 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Kolmogorov entropy is a well-suited measure for the complexity of dynamical systems containing noises. Approximate entropy (AppEn), proposed by Pincus [1], is an approximation of the Kolmogorov entropy. To overcome the biasedness of AppEn caused by self-matching, Richman proposed sample entropy (SampEn) [2] in 2000. SampEn is essentially defined as $-\log(B/A)$, where B denotes the number of matched template pairs with length m and A denotes the number of matched template pairs with $m+1$. SampEn has prevailed in many areas, such as cyber-physical systems, mechanical systems, health monitoring, disease diagnosis, and control. Based on AppEn and SampEn, multiscale entropy [3] and hierarchical entropy [4] were developed for measuring the complexity of physiological time series in multiple time scales. Since low-frequency filters are involved, multiscale entropy can weaken the influence of meaningless structures such as noise on complexity measurement. By adding the sample entropy of the high-frequency component of the time series, the hierarchical entropy provides more comprehensive and accurate information and improves the ability to distinguish different time series. Multiscale entropy, hierarchical entropy, and their variants have been applied to various fields such as fault identification [5,6] and feature extraction [7], beyond physiological time series analysis.

Computing SampEn requires counting the number of similar templates of time series. In other words, it requires counting the number of matched template pairs for a given time series. Clearly, direct computing of SampEn requires computational complexity of $O(N^2)$, where N is the length of the time series analyzed. To accelerate the computation of SampEn, kd-tree based algorithms for sample entropy were proposed, which reduce the

time complexity to $O(N^{2-\frac{1}{m+1}})$, where m is the template (also called pattern) length [8,9]. In addition, box-assisted [10,11], bucket-assisted [12], lightweight [13], and assisted sliding box (SBOX) [14] algorithms were developed. However, the complexity of all these algorithms is $O(N^2)$. Recently, an algorithm proposed in [15] for computing approximate values of AppEn and SampEn, without theoretical error analysis, still requires $O(N^2)$ computational costs in the worst scenario, even though it requires only $O(N)$ number of operations in certain best cases. Developing fast algorithms for estimating SampEn is still of great interest.

The goal of this study is to develop a Monte-Carlo-based algorithm for calculating SampEn. The most costly step in computing SampEn is to compute the matched template ratio B/A of length m over length $m+1$. Noting that $\frac{A}{N(N-1)}$ (resp. $\frac{B}{N(N-1)}$) is the probability that templates of length m (resp. $m+1$) are matched, the ratio B/A can be regarded as a conditional probability. From this viewpoint, we can approximate this conditional probability of the original data set by that of a data set randomly down-sampled from the original one. Specifically, we randomly select N_0 templates of lengths m and N_0 templates of $m+1$ from the original time series. We then count the number $\tilde{A}$ (resp. $\tilde{B}$) of matched pairs among the selected templates of lengths m (resp. $m+1$). We repeat this process N_1 times, and compute the mean $\bar{A}_{N_1}$ (resp. $\bar{B}_{N_1}$) of $\tilde{A}$ (resp. $\tilde{B}$). Then, we use $-\log(\bar{B}_{N_1}/\bar{A}_{N_1})$ to approximate $-\log(B/A)$ for the time series to measure its complexity. We establish the computational complexity and convergence rate of the proposed algorithm. We then study the performance of the proposed algorithm, by comparing it with the kd-tree-based algorithm and the SBOX method on the electrocardiogram (ECG) time series, electroencephalogram time series (EEG), cardiac inter-beat (RR) time series, mechanical vibration signals (MVS), meteorological data (MD), and $1/f$ noise. Numerical results show that the proposed algorithm can gain more than 100 times speedup compared to the SBOX algorithm (the most recent algorithm in the literature to the best of our knowledge) for a time series of length $2^{16}-2^{18}$, and more than 1000 times speedup for a time series of length $2^{19}-2^{20}$. Compared to the kd-tree algorithm, the proposed algorithm can again achieve up to 1000 times speedup for a time series of length 2^{20}.

This article is organized in five sections. The proposed Monte-Carlo-based algorithm for estimating sample entropy is described in Section 2. Section 3 includes the main results of the analysis of approximate accuracy of the proposed algorithm, and the proofs are given in the Appendix A. Numerical results are presented in Section 4, and conclusion remarks are made in Section 5.

2. Sample Entropy via Monte Carlo Sampling

In this section, we describe a Monte-Carlo-based algorithm for estimating the sample entropy of a time series.

We first recall the definition of sample entropy. For all $k \in \mathbb{N}$, let $\mathbb{Z}_k := \{1, 2, \ldots, k\}$. The distance of two real vectors $\mathbf{a} := [a_l : l \in \mathbb{Z}_k]$ and $\mathbf{b} := [b_l : l \in \mathbb{Z}_k]$ of length k is defined by

$$\rho(\mathbf{a}, \mathbf{b}) := \max\{|a_l - b_l| : l \in \mathbb{Z}_k\}.$$

We let $\mathbf{u} := (u_i \in \mathbb{R} : i \in \mathbb{Z}_n)$ be a time series of length $n \in \mathbb{N}$. For $m \in \mathbb{N}$, we let $N := n - m - 1$. We define a set X of N vectors by $X := \{\mathbf{x}_i : i \in \mathbb{Z}_N\}$, where $\mathbf{x}_i := [u_{i+l-1} : l \in \mathbb{Z}_m]$ is called a *template* of length m for the time series $\mathbf{u}$. We also define a set Y of N vectors by $Y := \{\mathbf{y}_i : i \in \mathbb{Z}_N\}$, where $\mathbf{y}_i := [u_{i+l-1} : l \in \mathbb{Z}_{m+1}]$ is called a *template* of length $m+1$ for $\mathbf{u}$. To avoid confusion, we call the elements in X and Y the templates for the time series $\mathbf{u}$. We denote by $\#E$ the cardinality of a set E. We use A_i, $i \in \mathbb{Z}_N$, to denote the cardinality of the set consisting of templates $\mathbf{x} \in X \setminus \{\mathbf{x}_i\}$ satisfying $\rho(\mathbf{x}_i, \mathbf{x}) \le r$, that is,

$$A_i := \#\{\mathbf{x} \in X \setminus \{\mathbf{x}_i\} : \rho(\mathbf{x}_i, \mathbf{x}) \le r\}.$$

Likewise, for $i \in \mathbb{Z}_N$, we let

$$B_i := \#\{\mathbf{y} \in Y \setminus \{\mathbf{y}_i\} : \rho(\mathbf{y}_i, \mathbf{y}) \le r\}.$$

Letting

$$B := \frac{1}{2}\sum_{i\in\mathbb{Z}_N} B_i \text{ and } A := \frac{1}{2}\sum_{i\in\mathbb{Z}_N} A_i,$$

we define the sample entropy of time series $\mathbf{u}$ by

$$\mathrm{SampEn}(\mathbf{u}, m, r) := \begin{cases} -\log\left(\frac{B}{A}\right), & \text{if } A > 0, B > 0, \\ -\log\left(\frac{2}{N(N-1)}\right), & \text{otherwise.} \end{cases}$$

The definition of sample entropy yields the direct algorithm, which explicitly utilizes two nested loops, where the inner one computes A_i and B_i, and the outer one computes A and B. Algorithm 1 will be called repeatedly in the Monte-Carlo-based algorithm to be described later.

Algorithm 1 Direct method for range counting

Require: Sequence $\mathbf{u} := (u_i : i \in \mathbb{Z}_{N+m})$, subset $\mathbf{s} \subset \mathbb{Z}_N$, template length m and threshold r.

```
procedure DirectRangeCounting(u, s, m, r)
    Set count = 0,
    Set L = #s,
    for i = 1 to L do
        Set a = [u_{s_i+l-1} : l ∈ Z_m],
        for j = i + 1 to L do
            Set b = [u_{s_j+l-1} : l ∈ Z_m],
            if ρ(a − b) ≤ r then
                count = count + 1,
    return count
```

The definition of sample entropy shows that sample entropy measures the predictability of data. Precisely, in the definition of sample entropy, B/A measures a conditional probability that when the distance of two templates $\mathbf{a}$ and $\mathbf{b}$ is less than or equal to r, the distance of their corresponding $(m+1)$-th component is also less than or equal to r. From this perspective, we can approximate this conditional probability of the original data set by computing it on a data set randomly down-sampled from the original one. To describe this method precisely, we define the notations as follows.

We choose a positive integer N_0, randomly draw N_0 numbers from $\mathbb{Z}_N$ without replacement, and form an N_0-dimensional vector. All of such vectors form a subset Ω of the product space

$$\mathbb{Z}_N^{N_0} := \mathbb{Z}_N \otimes \mathbb{Z}_N \otimes \cdots \otimes \mathbb{Z}_N \ \ (N_0\text{-folds}),$$

that is,

$$\Omega := \{\mathbf{s} := [s_1, \ldots, s_{N_0}] \in \mathbb{Z}_N^{N_0} : s_i \neq s_j \text{ for all } i \neq j\}.$$

Suppose that $\mathcal{F}$ is the power set of Ω (the set of all subsets of Ω, including the empty set and Ω itself). We let P be the uniform probability measure satisfying $P(\mathbf{s}) = 1/(\#\Omega)$ for all $\mathbf{s} \in \Omega$ and define the probability space $\{\Omega, \mathcal{F}, P\}$. The definition of Ω implies $\#\Omega = \frac{N!}{(N-N_0)!}$, and thus the probability measure satisfies $P(\mathbf{s}) = \frac{(N-N_0)!}{N!}$ for all $\mathbf{s} \in \Omega$. The definition of $\mathcal{F}$ means all events that may occur in the sample space Ω are considered in the probability space $\{\Omega, \mathcal{F}, P\}$. We randomly select N_0 templates of length m and N_0 templates of length $m+1$ from the original time series. We then count the number $\tilde{A}$ (resp. $\tilde{B}$) of matched pairs among the selected templates of lengths m (resp. $m+1$). That is,

$$\tilde{A}(\mathbf{s}) := \frac{1}{2}\#\Big\{(i,j) : i,j \in \mathbb{Z}_{N_0} \text{ with } i \neq j, \text{ and } \rho(\mathbf{x}_{s_i}, \mathbf{x}_{s_j}) \leq r\Big\}, \quad \mathbf{s} \in \Omega,$$

and

$$\tilde{B}(\mathbf{s}) := \frac{1}{2}\#\Big\{(i,j) : i,j \in \mathbb{Z}_{N_0} \text{ with } i \neq j, \text{ and } \rho(\mathbf{y}_{s_i}, \mathbf{y}_{s_j}) \leq r\Big\}, \quad \mathbf{s} \in \Omega.$$

We repeat this process N_1 times.

Note that $\tilde{A}$ and $\tilde{B}$ are random variables on the probability space $\{\Omega, \mathcal{F}, P\}$. Let $\bar{A}_{N_1}$ and $\bar{B}_{N_1}$ be the averages of random variables $\tilde{A}$ and $\tilde{B}$, respectively, over the N_1 repeated processes. That is,

$$\bar{A}_{N_1} := \frac{1}{N_1}\sum_{k=1}^{N_1} \tilde{A}(\mathbf{s}_k), \quad \text{and} \quad \bar{B}_{N_1} := \frac{1}{N_1}\sum_{k=1}^{N_1} \tilde{B}(\mathbf{s}_k),$$

where $\{\mathbf{s}_k : k \in \mathbb{Z}_{N_1}\}$ is a subset of Ω. With $\bar{A}_{N_1}$ and $\bar{B}_{N_1}$, we can estimate the sample entropy $-\log(B/A)$ by computing $-\log(\bar{B}_{N_1}/\bar{A}_{N_1})$. We summarize the procedure for computing $-\log(\bar{B}_{N_1}/\bar{A}_{N_1})$ in Algorithm 2 and call it *the Monte-Carlo-based algorithm for evaluating sample entropy* (MCSampEn). In MCSampEn, $\mathbf{s}_k$, $k \in \mathbb{Z}_{N_0}$, are selected by the Hidden Shuffle algorithm proposed in [16].

Algorithm 2 Monte-Carlo-based algorithm for evaluating sample entropy

Require: Sequence $\mathbf{u} = (u_i : i \in \mathbb{Z}_{N+m})$, template length m, tolerance $r \in \mathbb{R}$, sample size N_0 and number of experiments N_1, probability space $\{\Omega, \mathcal{F}, P\}$

1: **procedure** MCSAMPEN($\mathbf{u}, m, r, N_0, N_1$)
2: Set $\bar{A}_{N_1} = 0$ and $\bar{B}_{N_1} = 0$,
3: **for** $k = 1$ to N_1 **do**
4: Select $\mathbf{s}_k \in \Omega$, randomly, with uniform distribution,
5: Compute $\tilde{A}(\mathbf{s}_k)$ by calling DirectRangeCounting($\mathbf{u}, \mathbf{s}^{(k)}, m, r$),
6: Compute $\tilde{B}(\mathbf{s}_k)$ by calling DirectRangeCounting($\mathbf{u}, \mathbf{s}^{(k)}, m+1, r$),
7: $\bar{A}_{N_1} = \bar{A}_{N_1} + \frac{1}{N_1}\tilde{A}(\mathbf{s}^{(k)})$,
8: $\bar{B}_{N_1} = \bar{B}_{N_1} + \frac{1}{N_1}\tilde{B}(\mathbf{s}^{(k)})$,
9: *entropy* $= -\log \frac{\bar{B}_{N_1}}{\bar{A}_{N_1}}$,
10: **return** *entropy*

We next estimate the computational complexity of MCSampEn measured by the number of arithmetic operations. To this end, we recall Theorem 3.5 of [16] which gives the number of arithmetic operations used in the Hidden Shuffle algorithm.

Theorem 1. *The Hidden Shuffle algorithm generates a random sample of size N_0 sequentially from a population of size N with $O(N_0)$ arithmetic operations in total.*

Theorem 2. *The total number of arithmetic operations needed in Algorithm 2 is $O(N_1(N_0^2 + N_0))$.*

Proof. For each $k \in \mathbb{Z}_{N_1}$, according to Theorem 1, the number of arithmetic operations needed for selecting $\mathbf{s}^{(k)}$ on line 4 of Algorithm 2 is $O(N_0)$. Moreover, from Algorithm 1 we can see that for each $k \in \mathbb{Z}_{N_1}$, the number of arithmetic operations needed for computing $\tilde{A}(\mathbf{s}_k)$ and $\tilde{B}(\mathbf{s}_k)$ on lines 5 and 6 is $O(N_0^2)$. Thus, by counting the number of arithmetic operations needed for lines 7, 8, and 9 of Algorithm 2, we obtain the desired result. □

Theorem 2 indicates that the computational complexity of MCSampEn is controlled by setting appropriate sampling parameters N_0 and N_1. When N_0 and N_1 are fixed, the computational complexity of MCSampEn is independent of the length N of time series $\mathbf{u}$. Meanwhile, we can also select N_0 and N_1 depending on N to balance the error and computational complexity of MCSampEn. For example, we can set $N_0 := \max\{1024, \lfloor\sqrt{N}\rfloor\}$ and $N_1 := \min\{5 + \log_2 N, \lfloor N/N_0\rfloor\}$, where $\lfloor a\rfloor$ denotes the greatest integer no bigger than $a \in \mathbb{R}$. In this case, the computational complexity is $O(N \log_2 N)$.

Noting that MCSampEn provides an approximation of the sample entropy, and not the exact value, convergence of MCSampEn is an important issue. We will discuss this in Section 3.

3. Error Analysis

In this section, we analyze the error of MCSampEn. Specifically, we will establish an approximation rate of MCSampEn in the sense of almost sure convergence.

A sequence of $\{V_k : k \in \mathbb{N}\}$ of random variables in probability space $\{\Omega, \mathcal{F}, P\}$ is said to converge almost surely to $V \in \{\Omega, \mathcal{F}, P\}$, denoted by

$$V_k \xrightarrow{a.s.} V,$$

if there exists a set $\mathcal{N} \in \mathcal{F}$ with $P(\mathcal{N}) = 0$ such that for all $\omega \in \Omega \setminus \mathcal{N}$,

$$\lim_{k\to\infty} V_k(\omega) = V(\omega).$$

It is known (see [17]) that $\{V_k : k \in \mathbb{N}\}$ converges almost surely to $V \in \{\Omega, \mathcal{F}, P\}$ if and only if

$$\lim_{k\to+\infty} P\left(\left\{\sup_{i\geq k}|V_i - V| > \epsilon\right\}\right) = 0, \text{ for all } \epsilon > 0.$$

Furthermore, we can describe the convergence rate of $\{V_i : i \in \mathbb{N}\}$ by the declining rate of the sequence $\left\{P\left(\left\{\sup_{i\geq k}|V_i - V| > \epsilon\right\}\right) : k \in \mathbb{N}\right\}$ for all $\epsilon > 0$. If for $\alpha > 0$,

$$P\left(\left\{\sup_{i\geq k}|V_i - V| > \epsilon\right\}\right) = O(k^{-\alpha}), \text{ for all } \epsilon > 0,$$

we say $\{V_i : i \in \mathbb{N}\}$ converges to V almost surely with rate α.

To establish the approximation error of MCSampEn, we first derive two theoretical results for the expectations and variations of $\frac{\tilde{A}}{N_0(N_0-1)}$ and $\frac{\tilde{B}}{N_0(N_0-1)}$. Then, by combining these results with the results of the almost sure convergence rate in [18] and the local smoothness of logarithm functions, we obtain the approximation rate of $\{-\log(\tilde{B}_{N_1}/\tilde{A}_{N_1}) : N_1 \in \mathbb{N}\}$ in the sense of almost sure convergence, which is the main theoretical result of this paper. We state these results below and postpone their proofs to the Appendix A.

The expectations of $\frac{\tilde{A}}{N_0(N_0-1)}$ and $\frac{\tilde{B}}{N_0(N_0-1)}$ are given in the following theorem.

Theorem 3. *It holds that for all $N_0 \in \mathbb{Z}_N$ with $N_0 > 1$,*

$$\mathrm{E}\left[\frac{\tilde{A}}{N_0(N_0-1)}\right] = \frac{A}{N(N-1)}, \tag{1}$$

and

$$\mathrm{E}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right] = \frac{B}{N(N-1)}. \tag{2}$$

The next theorem presents the variations of $\frac{\tilde{A}}{N_0(N_0-1)}$ and $\frac{\tilde{B}}{N_0(N_0-1)}$.

Theorem 4. *It holds that for all $N_0 \in \mathbb{Z}_N$ with $N_0 > 1$,*

$$\mathrm{Var}\left[\frac{\tilde{A}}{N_0(N_0-1)}\right] = \frac{C_{N_0}}{N_0}, \tag{3}$$

and

$$\mathrm{Var}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right] = \frac{C_{N_0}}{N_0}, \tag{4}$$

where

$$C_{N_0} := \frac{B}{(N_0-1)N(N-1)} + \frac{N_0-2}{(N_0-1)N(N-1)(N-2)}\left(\sum_{l=1}^{N} B_l^2 - 2B\right) + \frac{(N_0-2)(N_0-3)}{(N_0-1)N(N-1)(N-2)(N-3)}\left(B^2 - \sum_{l=1}^{N} B_l^2 + B\right) - \frac{N_0 B^2}{N^2(N-1)^2}. \quad (5)$$

Moreover, there is $0 < C_{N_0} < 1 + \frac{1}{2(N_0-1)}$.

Based on Theorems 3 and 4, we can obtain $\log \frac{\bar{B}_{N_1}}{\bar{A}_{N_1}} \xrightarrow{a.s.} \log \frac{B}{A}$ by the Kolmogorov strong law of large numbers and the continuous mapping theorem. However, in practice it is desirable to quantify the approximation rate in the sense of almost sure convergence, so that we can estimate the error between $\log \frac{\bar{B}_{N_1}}{\bar{A}_{N_1}}$ and $\log \frac{B}{A}$. To this end, we define $\tau_A := \mathrm{E}\left[\left|\frac{\tilde{A}}{N_0(N_0-1)} - \frac{A}{N(N-1)}\right|\right]$, and $\tau_B := \mathrm{E}\left[\left|\frac{\tilde{B}}{N_0(N_0-1)} - \frac{B}{N(N-1)}\right|\right]$. Let $\gamma_A := \frac{A}{2N(N-1)e}$ and $\gamma_B := \frac{B}{2N(N-1)e}$. For all $\beta > 1$ and $0 < \epsilon < 1$, we also let

$$n_{\epsilon,\beta} := \max\left\{6\epsilon^{-1}, \exp\left(\left(9\beta^{-1}\epsilon^{-1}\right)^{\beta^{-1}/\left(1-\beta^{-1}\right)}\right)\right\}. \quad (6)$$

With the notation defined above, we present below the main theoretical result of this paper, which gives the rate of $\{-\log \frac{\bar{B}_k}{\bar{A}_k} : k \in \mathbb{N}\}$ approximating $-\log \frac{B}{A}$ in the sense of almost sure convergence.

Theorem 5. *Let* $\beta > 1$ *and* $N_0 \in \mathbb{Z}_N$ *with* $N_0 > 3$. *If* $A, B > 0$, *then there exist constants* D_β *and* $\tilde{D}_\beta$ *(depending only on* β*) such that for all* $0 < \epsilon < 1$ *and* $N_1 > n_{\epsilon,\beta}$, *such that*

$$P\left(\sup_{k>N_1}\left|\log\frac{\bar{B}_k}{\bar{A}_k} - \log\frac{B}{A}\right| > \max\{\tau_A, \tau_B\}\epsilon\right) \leq \frac{72C_{N_0}}{\epsilon^2 N_0 N_1}\left(D_\beta + \tilde{D}_\beta(\log N_1)^{\beta-1}\right)\left(\frac{1}{\tau_A^2\gamma_A^2} + \frac{1}{\tau_B^2\gamma_B^2}\right). \quad (7)$$

The proof for Theorems 3–5 are included in the Appendix A. Note that Theorem 5 indicates that $-\log \frac{\bar{B}_k}{\bar{A}_k}$ approximates $-\log \frac{B}{A}$ in the sense of almost sure convergence of order 1.

4. Experiments

We present numerical experiments to show the accuracy and computational complexity of the proposed algorithm MCSampEn.

As sample entropy has been prevalently used in a large number of areas, we consider several series with a variety of statistical features, including the electrocardiogram (ECG) series, RR interval series, electroencephalogram (EEG) series , mechanical vibration signals (MVS), meteorological data (MD), and $1/f$ noise. The ECG and EEG data can be downloaded from PhysioNet, a website offering access to recorded physiologic signals (PhysioBank) and related open-source toolkits (PhysioToolkit) [19]. The MVS data can be found in [20] and the website of the Case Western Reserve University Bearing Data Center [21]. The MD data can be downloaded from the website of the Royal Netherlands Meteorological Institute [22]. The databases used in this paper include:

Long-Term AF Database (ltafdb) [23]. This database includes 84 long-term ECG recordings of subjects with paroxysmal or sustained atrial fibrillation (AF). Each record

contains two simultaneously recorded ECG signals digitized at 128 Hz with 12-bit resolution over a 20 mV range; record durations vary but are typically 24 to 25 h.

Long-Term ST Database (ltstdb) [24]. This database contains 86 lengthy ECG recordings of 80 human subjects, chosen to exhibit a variety of events of ST segment changes, including ischemic ST episodes, axis-related non-ischemic ST episodes, episodes of slow ST level drift, and episodes containing mixtures of these phenomena.

MIT-BIH Long-Term ECG Database (ltecg) [19]. This database contains 7 long-term ECG recordings (14 to 22 h each), with manually reviewed beat annotations.

BIDMC Congestive Heart Failure Database (chfdb) [25]. This database includes long-term ECG recordings from 15 subjects (11 men, aged 22 to 71, and 4 women, aged 54 to 63) with severe congestive heart failure (NYHA class 3–4).

MGH/MF Waveform Database (mghdb) [26]. The Massachusetts General Hospital/ Marquette Foundation (MGH/MF) Waveform Database is a comprehensive collection of electronic recordings of hemodynamic and electrocardiographic waveforms of stable and unstable patients in critical care units, operating rooms, and cardiac catheterization laboratories. Note that only the ECG records were considered in our experiments.

RR Interval Time Series (RR). The RR interval time series are derived from healthy subjects (RR/Health), and subjects with heart failure (RR/CHF) and atrial fibrillation (RR/AF).

CHB-MIT Scalp EEG Database (chbmit) [27]. This database contains (EEG) records of pediatric subjects with intractable seizures. The records are collected from 22 subjects, monitored for up to several days.

Gearbox Database (gearbox) [20]. The gearbox dataset was introduced in [20] and was published on https://github.com/cathysiyu/Mechanical-datasets (accessed on 27 March 2022).

Rolling Bearing Database (RB) [21]. This database as a standard reference for the rolling bearing fault diagnosis is provided by the Case Western Reserve University Bearing Data Center [21].

Meteorological Database (MD) [22]. The meteorological database used in this section records the hourly weather data in the past 70 years in the Netherlands.

As each database consists of multiple records from different subjects, we select one record randomly from each database. Specifically, we choose record "00" from ltafdb, "s20011" from ltstdb, "14046" from ltdb, "chf01" from chfdb, "mgh001" from mghdb, "chb07_01" from chbmit, "Miss_30_2" from gearbox, "XE110_DE_Time" from RB, and "380_t" from MD. Moreover, $1/f$ noise signal, an artificial signal, is studied to increase diversity. The time series considered in this section are illustrated in Figure 1, where all samples are normalized to have a standard deviation of 1, since the parameter threshold r is proportional to the standard deviation of the records, and thus the whole range of the records is negligible.

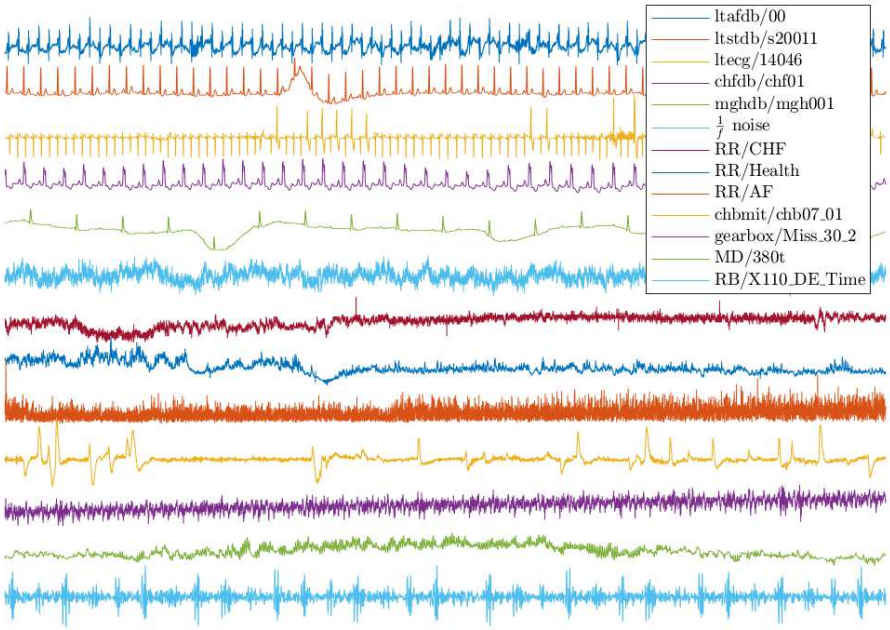

Figure 1. Samples of the dataset records.

4.1. Approximation Accuracy

In the experiments presented in this subsection, we examine the approximation accuracy of the MCSampEn algorithm. Specifically, we set $r := 0.15$ and $m := 4, 5$. We vary the sampling size N_0 and the number N_1 of computations to study the approximation accuracy of the proposed algorithm. In this experiment, records with lengths exceeding 10^6 are truncated to have length 10^6; otherwise, the entire records are used. Since in the MCSampEn algorithm, $\mathbf{s}_k \in \Omega$ are selected randomly, the outcome of the algorithm depends on the selected value of $\mathbf{s}_k$. To overcome the effect of the randomness, for every specified pair of (N_0, N_1), we run the algorithm 50 times and calculate the mean errors **(MeanErr)** and the root mean squared errors **(RMeanSqErr)** of the 50 outcomes.

In our first experiment, we consider series "mghdb/mgh001", select parameters $N_0 \in \{200i : i \in \mathbb{Z}_{20}^+\}$, $N_1 \in \{10i : i \in \mathbb{Z}_{25}^+\}$, and show in Figure 2 the mean errors and the root mean squared errors of the MCSampEn outputs as surfaces in the N_0-N_1 coordinate system. Images (a) and (c) of Figure 2 show the values of *MeanErr* and images (b), (d), and (f) of Figure 2 show the values of *RMeanSqErr*. Figure 2 clearly demonstrates that both the mean errors and the root mean squared errors of the MCSampEn outputs converge to 0 as N_0 or N_1 increases to infinity. This is consistent with our theoretical analysis in the previous section.

In the second experiment, we consider all series illustrated in Figure 1 and show numerical results in Figures 3 and 4. Images (a), (c), and (e) of Figure 3 show the values of *MeanErr*, and images (b), (d), and (f) of Figure 3 show the values of *RMeanSqErr*, with $N_0 \in \{200i : i \in \mathbb{Z}_{20}^+\}$ and fixed $N_1 = 250$. Images (a), (c), and (e) of Figure 4 show the values of *MeanErr*, and images (b), (d), and (f) of Figure 4 show the values of *RMeanSqErr*, with $N_0 = 4000$ and $N_1 \in \{10i : i \in \mathbb{Z}_{25}^+\}$. Figure 3 indicates that the outputs of the MCSampEn algorithm converge as N_0 increases. We can also see from Figure 3 that when $N_0 \geq 1500$, $N_1 = 150$, and $m = 4$, both MeanErr and RMeanSqErr are less than 1×10^{-2} for all tested time series. In other words, the MCSampEn algorithm can effectively estimate sample entropy when $N_0 \geq 1500$, $N_1 = 150$, and $m = 4$. From Figure 4, we can also observe that the outputs of the MCSampEn algorithm converge as N_1 increases. This is consistent with the theoretical results established in Section 3.

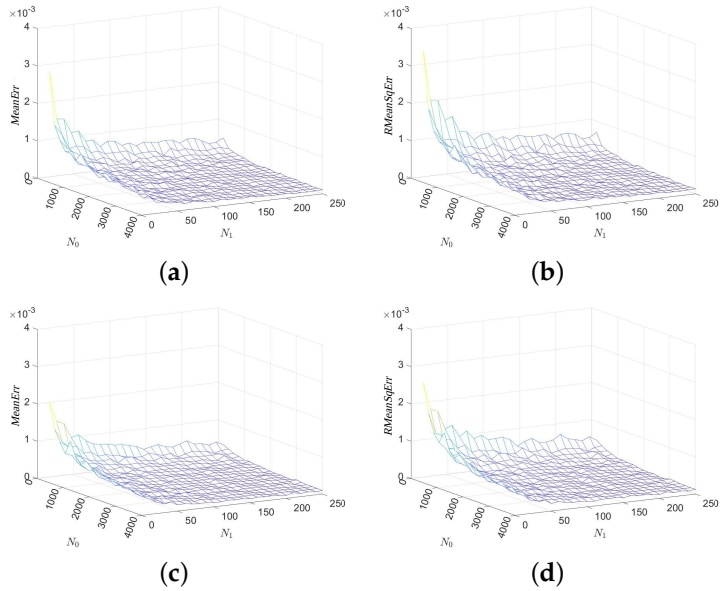

Figure 2. The values of *MeanErr* and *RMeanSqErr* for time series "mghdb/mgh001" with respect to the sample size N_0 and the number of computations N_1, where parameters $r = 0.15$ and $m = 4, 5$. (**a**) *MeanErr* with $m = 4$. (**b**) *RMeanSqErr* with $m = 4$. (**c**) *MeanErr* with $m = 5$. (**d**) *RMeanSqErr* with $m = 5$.

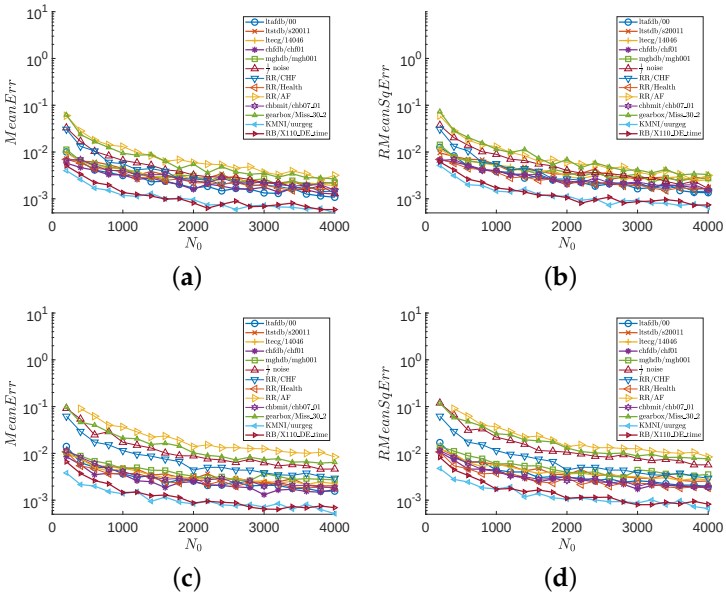

Figure 3. The values of *MeanErr* and *RMeanSqErr* with respect to $N_0 \in \{200i : i \in \mathbb{Z}_{20}^{+}\}$ and $N_1 = 150$, where parameters $r = 0.15$ and $m = 4, 5$. (**a**) *MeanErr* with $m = 4$. (**b**) *RMeanSqErr* with $m = 4$. (**c**) *MeanErr* with $m = 5$. (**d**) *RMeanSqErr* with $m = 5$.

We next explain how the randomness of a time series effects the accuracy of the MCSampEn algorithm by applying the algorithm to the stochastic process $MIX(p)$, which has been widely applied to studies of sample entropy [1,2,28]. The $MIX(p)$ is defined as follows. Let $x_j := \alpha^{-1/2} \sin(12\pi j/12)$ for all $j \in \mathbb{Z}_N$ where

$$\alpha := \left(\sum_{j=1}^{12} \sin^2(2\pi j/12) \right) /12.$$

Let $\{y_j : j \in \mathbb{Z}_N\}$ be a family of independent identically distributed (i.i.d) real random variables with uniform probability density on the interval $[-\sqrt{3}, \sqrt{3}]$. Note that $\{x_j : j \in \mathbb{Z}_N\}$ and $\{y_j : j \in \mathbb{Z}_N\}$ are sequences with contrary properties: the former is a completely regular sine sequence, and the latter is completely random. Let $p \in [0, 1]$, and $\{z_j : j \in \mathbb{Z}_N\}$ be a family of i.i.d random variables satisfying $z_j = 1$ with probability p and $z_j = 0$ with probability $1 - p$. Then, the $MIX(p)$ process is defined as $\{m_j := (1 - z_j)x_j + z_j y_j : j \in \mathbb{Z}_N\}$. It's not hard to find that the parameter p controls the ratio of sine sequence and random noise in the $MIX(p)$ process and the increase in p makes the $MIX(p)$ process more random. When $p = 0$, the $MIX(p)$ process is a deterministic sine sequence. Meanwhile, when $p = 1$, the $MIX(p)$ process turns out completely unpredictable uniform noise. This feature makes it an ideal series to study how randomness affects the accuracy of the MCSampEn algorithm.

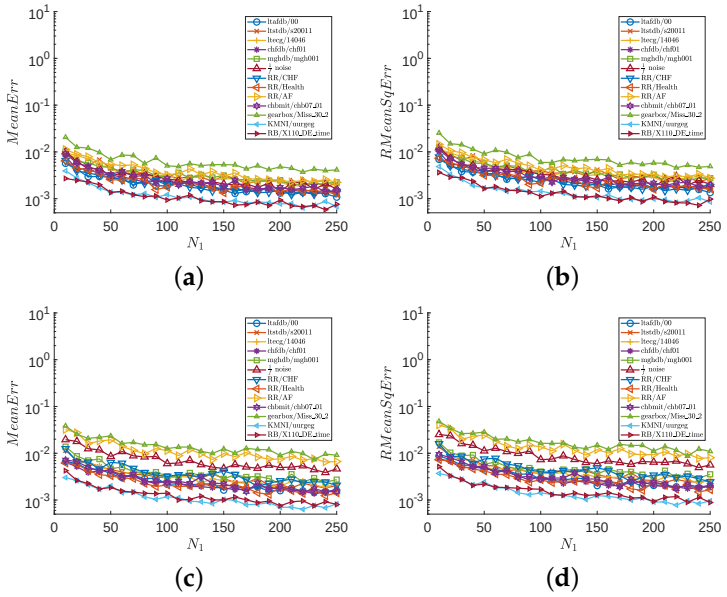

Figure 4. The values of *MeanErr* and *RMeanSqErr* with respect to $N_0 = 2 \times 10^3$ and $N_1 \in \{10i : i \in \mathbb{Z}_{25}^+\}$, where parameters $r = 0.15$ and $m = 4, 5$. (**a**) *MeanErr* with $m = 4$. (**b**) *RMeanSqErr* with $m = 4$. (**c**) *MeanErr* with $m = 5$. (**d**) *RMeanSqErr* with $m = 5$.

Here, we apply MCSampEn to $MIX(p)$, $p \in \{0.5 + 0.5i : i \in \mathbb{Z}_{19}\}$ and show the results of *RMeanSqErr* versus p in Figure 5. From Figure 5, we can observe that the values of *RMeanSqErr* increase linearly with a very small growth rate when $p \leq 0.5$. When $p > 0.5$, the values of *RMeanSqErr* are significantly faster than that of $p \leq 0.5$. Therefore, we believe that when the randomness of a time series is weak, the error of the MCSampEn algorithm is small; as the randomness of the time series increases, the error of the MCSampEn grows.

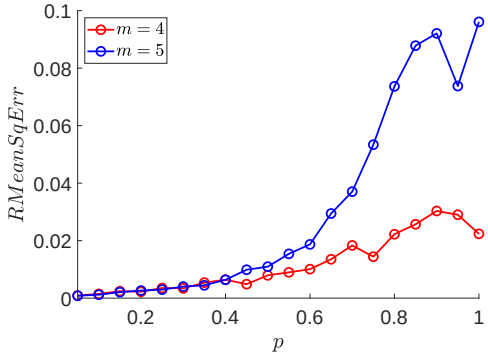

Figure 5. The values of *RMeanSqErr* with respect to p, where parameters $r = 0.15$, $m = 4, 5$, $N = 2^{20}$, $N_0 = 2000$, and $N_1 = 150$.

4.2. Time Complexity

In the experiments presented in this subsection, we compare the computing time of the MCSampEn algorithm with that of the kd-tree algorithm [8] and SBOX algorithm [14], under the condition that the value of sample entropy computed by the MCSampEn algorithm is very close to the ground truth value. The computational time experiments are performed on a desktop computer running Windows 11, with an Intel(R) Core(TM) i5-9500 CPU, and 32GB RAM. The implementations of the kd-tree-based algorithm and the MCSampEn algorithm are available on the website https://github.com/phreer/fast_sampen_impl.git (accessed on 30 March 2022). As for the SBOX method, we utilize the implementation given by the original author, published on website https://sites.google.com/view/yhw-personal-homepage (accessed on 25 October 2021). To demonstrate the validity of the MCSampEn algorithm, we also show both the sample entropy estimated by MCSampEn and the corresponding ground truth.

As we have discussed above, the time complexity of the MCSampEn algorithm depends on the parameters N_0 and N_1. In this subsection, we discuss two strategies for choosing N_0 and N_1:

S1 Choose N_0 and N_1 to be independent of N, for example $N_0 = 2 \times 10^3$ and $N_1 = 150$.

S2 Choose $N_0 = \max\{1024, \lfloor\sqrt{N}\rfloor\}$ and $N_1 = \min\{5 + \log_2 N, \lfloor N/N_0 \rfloor\}$, depending on N.

An intuitive explanation of the second strategy is shown below. We would like to choose N_0 and N_1 such that the overall time complexity of executing the algorithm is $O(N \log N)$. For this purpose, we expect N_0 to grow like $\sqrt{N}$ and N_1 to grow logarithmically in N. However, when N is not large enough, lack of sampling templates can seriously impair the accuracy of the algorithm. To overcome this problem, we set a lower bound of N_0 to 1024, which is a good trade-off between accuracy and time complexity. The experimental results in this subsection show that this strategy can produce satisfactory output even when N is small.

The results on different signals "ltafdb/00", "$1/f$ noise", "chbmit/chb07_01", and "ltecg/14046" are shown in Figure 6, where the first strategy is adopted by setting $N_0 = 2 \times 10^3$ and $N_1 = 150$, and the results for $m = 4$ are marked by red color, and the results for $m = 5$ are marked by blue. In the left column of Figure 6, the values of computation time consumed by the kd-tree, SBOX, and MCSampEn algorithms are plotted, respectively, with the dashed lines marked "x", the dash-dot lines marked "+", and the solid lines marked "o". From the results shown in the left column of Figure 6, we can find that MCSampEn is faster than the SBOX algorithm when N is greater than 2^{15}. We also can see when the time series "chbmit/chb07_01" and "ltecg/14046" have length N of 2^{20}, MCSampEn is nearly 1000 times faster than the SBOX algorithm. Compared to the kd-tree algorithm, the MCSampEn algorithm can still achieve up to hundreds of times acceleration when $N = 2^{20}$. In addition, the time complexity of MCSampEn algorithm is close to a constant relative to m, and is much smaller than the kd-tree and SBOX algorithms when N is large enough. Meanwhile, the computational time (shown in the left column of Figure 6) required is hardly affected by the times series length N.

The right column of Figure 6 shows the average of 50 outputs of the MCSampEn algorithm for different time series under the settings of $N_0 = 2 \times 10^3$ and $N_1 = 150$, where the red solid lines plot the average for the cases of $m = 4$, and the blue solid lines plot the average for the cases of $m = 5$. In the right column of Figure 6, the values of ground truth for the cases of $m = 4$ and $m = 5$ are plotted by the red and blue dashed lines, respectively. Meanwhile, in the right column of Figure 6, we use error bars "I" to represent the values of $RMeanSqErr$, where the larger the value of $RMeanSqErr$, the longer the error bar "I". From the length of error bar "I", we can see that the values of $RMeanSqErr$ are small compared to the ground truth. Especially on the time series "ltafdb/00", "chbmit/chb_0701", and "ltecg/14046", the values of $RMeanSqErr$ are negligible compared to the values of ground

truth. These results imply that when $N_0 = 2 \times 10^3$ and $N_1 = 150$, the sample entropy estimated by the MCSampEn algorithm can effectively approximate the ground truth value.

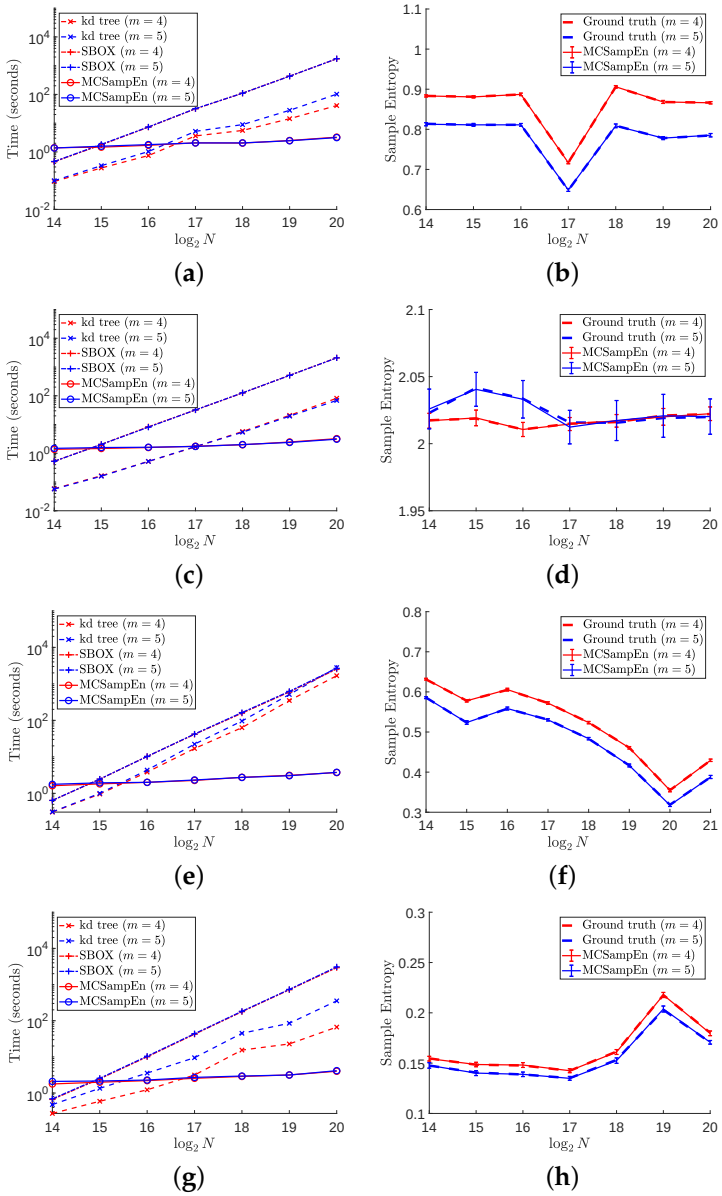

Figure 6. The left column shows the results of computational time versus data length N on different signals. In the right column, the values of *RMeanSqErr* are presented by error bars "I", where the larger the value of *RMeanSqErr*, the longer the error bar "I". In this figure, we set $m = 4, 5$, $N_0 = 2 \times 10^3$, and $N_1 = 150$. (**a**) Time for "ltafdb/00". (**b**) Sample entropy "ltafdb/00". (**c**) Time for $1/f$ noise. (**d**) Sample entropy for $1/f$ noise. (**e**) Time for "chbmit/chb07_01". (**f**) Sample entropy for "chbmit/chb07_01". (**g**) Time "ltecg/14046". (**h**) Sample entropy for "ltecg/14046".

The results of the second strategy are shown in Figure 7, where $N_0 = \max\{1024, \lfloor\sqrt{N}\rfloor\}$ and $N_1 = \min\{5 + \log_2 N, \lfloor N/N_0 \rfloor\}$. The results for $m = 4$ are marked by red color, and the results for $m = 5$ are marked by blue color. The left column of Figure 6 shows the values of computation time consumed by the kd-tree, SBOX, and MCSampEn algorithms, which are presented by the dashed lines marked "x", the dash-dot lines marked "+", and the solid lines marked "o", respectively. From the left column of Figure 7, we also can see that with the second strategy, the computational time of MCSampEn algorithm is much less than that of the kd-tree and SBOX algorithms, since the computational complexity of

Algorithm 2 is $O(N \log N)$. Furthermore, we observe that MCSampEn achieves a speedup of more than 100 compared to the SBOX algorithm when N goes from 2^{16} to 2^{18}, and it is over 1000 times faster when $N = 2^{20}$. Compared to the kd-tree algorithm, the MCSampEn algorithm can still obtain up to 1000 times acceleration when $N = 2^{20}$.

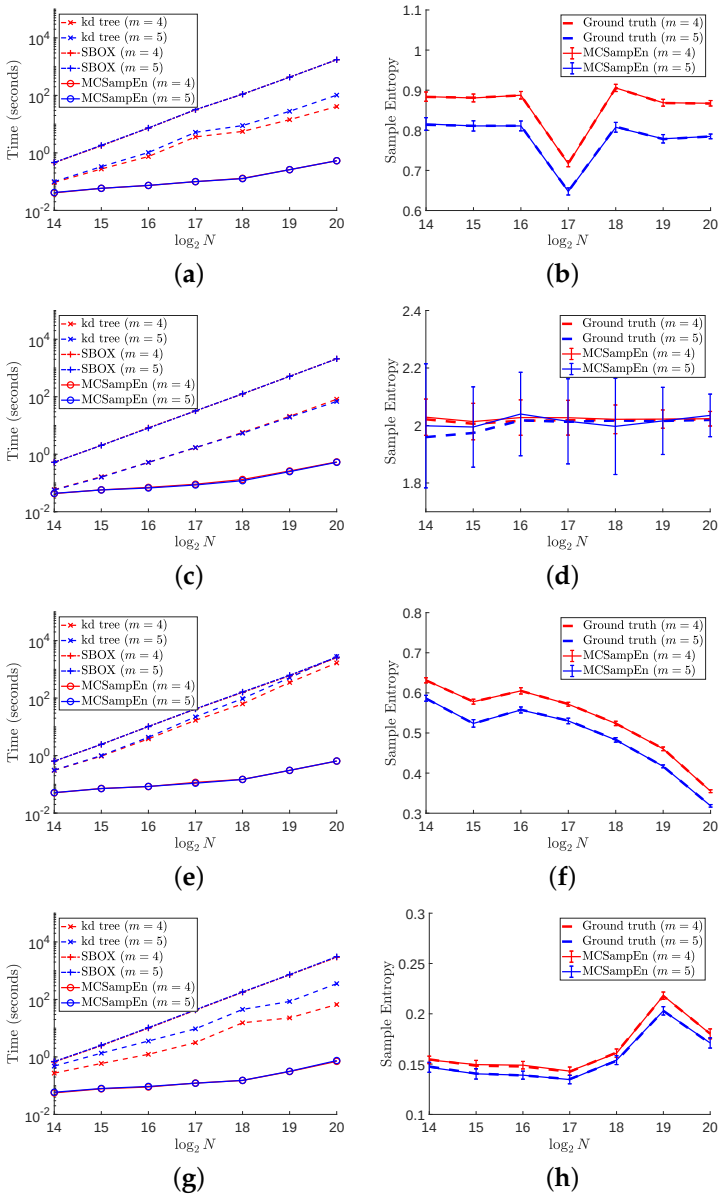

Figure 7. The left column shows the results of computational time versus data length N on different signals. The right column shows the values of *RMeanSqErr* by error bar, where the larger the value of *RMeanSqErr*, the longer the error bar "I". In this figure, we set $m = 4, 5$, $N_0 = \max\{1024, \lfloor\sqrt{N}\rfloor\}$, and $N_1 = \max\{1, \lfloor N/N_0 \rfloor\}$. (**a**) Time for "ltafdb/00". (**b**) Sample entropy "ltafdb/00". (**c**) Time for $1/f$ noise. (**d**) Sample entropy for $1/f$ noise. (**e**) Time for "chbmit/chb07_01". (**f**) Sample entropy for "chbmit/chb07_01". (**g**) Time "ltecg/14046". (**h**) Sample entropy for "ltecg/14046".

In the right column of Figure 7, we plot the average of 50 outputs of the MCSampEn algorithm for different time series by the red and blue solid lines for $m = 4$ and $m = 5$, respectively. At the same time, the values of ground truth for the cases of $m = 4$ and $m = 5$ are plotted by the red and blue dashed lines, respectively. As in Figure 6, we use the error bar "I" to represent the values of *RMeanSqErr*. Comparing the error bar "I" in Figure 6, we can see that the values of the *RMeanSqErr* in this experiment are larger than that shown in

Figure 6. However, the value of *RMeanSqErr* is still small in terms of the values of ground truth. Moreover, we can observe that the length of the error bars decreases as N increases. This means that we can obtain a better approximation of sample entropy as the time series length increases.

To reveal the effect of randomness on the speedup, we compare the time taken by the kd-tree and MCSampEn algorithms to compute the sample entropy of the time series $MIX(p)$, $p \in \{0.5 + 0.5i : i \in \mathbb{Z}_{19}\}$. The experimental results are shown in Figure 8, where the results for $m = 4$ are marked by red color, and the results for $m = 5$ are marked by blue. The values of computation time consumed by the kd-tree and MCSampEn algorithms are plotted, respectively, with the dashed lines marked "x" and the solid lines marked "o". In this experiment, we set $N = 2^{20}$ and $r = 0.15$. We also let $N_0 = 1000 + 3000p$ and $N_1 = 80 + 70p$ to ensure that the relative error $RMeanSqErr/SampEn$ is no greater than 0.02. From Figure 8, we can see that when the value of p is less than 0.2, compared with the kd-tree algorithm, the MCSampEn algorithm can achieve 300 to 1000 times speedup. When the value of p is greater than 0.8, our algorithm can still obtain a 10x speedup relative to the kd-tree algorithm.

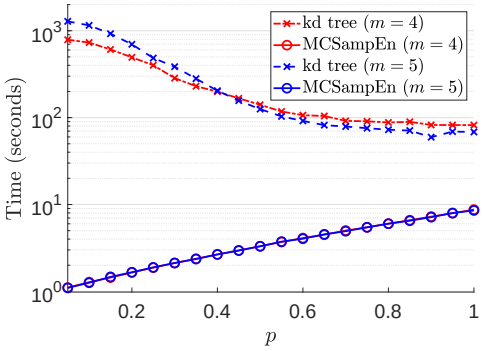

Figure 8. The results of computational time with respect to p, where parameters $r = 0.15$, $m = 4, 5$, $N = 2^{20}$, N_0, and N_1 are selected such that relative error $RMeanSqErr/SampEn \leq 0.02$.

From the experiments in this subsection, we can observe that the MCSampEn algorithm can achieve a high speedup when it is applied to different types of signals. In fact, compared with kd-tree algorithm, the MCSampEn algorithm can achieve high accuracy and more than 300 times acceleration when the time series has less randomness. When the randomness of the time series is high, our algorithm can still obtain a speedup of nearly 10 times.

4.3. Memory Usage

In order to show the performance of the MCSampEn algorithm more comprehensively, we also compare the memory usage of the kd-tree and MCSampEn algorithms. The memory usage on signal "ltstdb/s20011" is shown in Figure 9, where the memory usage for $m = 4$ and $m = 5$ is shown in Figure 9a,b, respectively. In this figure, the memory usage of the kd-tree algorithm is plotted by the blue dash-dot lines marked "x". The memory usage of the MCSampEn algorithm with the first and second strategies is plotted by the green dashed lines marked "+" and the red dotted lines marked "o", respectively. In Figure 9, the first strategy is adopted by setting $N_0 = 2048$ and $N_1 = 150$, and the second strategy is adopted by $N_0 = \max\{1024, \lfloor\sqrt{N}\rfloor\}$ and $N_1 = \min\{5 + \log_2 N, \lfloor N/N_0\rfloor\}$. We also present the memory usage for storing the data by the black solid lines marked "□".

From the results shown in Figure 9, it can be seen that when the size of the data is 2^{20}, the memory required by the kd-tree algorithm is almost 36 times that of the memory required by the MCSampEn algorithm. This is because the kd-tree algorithm requires a large memory space to save the kd-tree. Meanwhile, the experimental results in Figure 9 also show that the amount of memory required by the MCSampEn algorithm is only about

15 MB more than the amount of memory required to store the data when the length of data is between 2^{14} and 2^{24}. This is because the MCSampEn algorithm requires additional memory for storing N_0 templates and to execute the subroutines that generate random numbers.

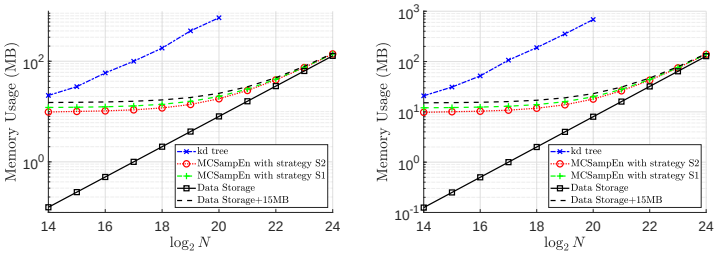

Figure 9. The results of memory usage versus data length N with $m = 4, 5$. (**a**) Memory usage for $m = 4$. (**b**) Memory usage for $m = 5$.

Because the MCSampEn algorithm is based on Monte Carlo sampling and the law of large numbers, it is an easily parallelizable algorithm. Therefore, combined with distributed storage techniques, the idea of the MCSampEn algorithm can be used to compute sample entropy for large-scale data (for example, where the size of data is larger than 1 TB). Parallel algorithms for computing sample entropy of large-scale data will be our future work.

5. Conclusions

In this paper, we propose a Monte-Carlo-based algorithm called MCSampEn to estimate sample entropy and prove that the outputs of MCSampEn can approximate sample entropy in the sense of almost sure convergence of order 1. We provide two strategies to select the sampling parameters N_0 and N_1, which appear in MCSampEn. The experiment results show that we can flexibly select the parameters N_0 and N_1 to balance the computational complexity and error. From the experimental results, we can observe that the computational time consumed by the proposed algorithm is significantly shorter than the kd-tree and SBOX algorithms, with negligible loss of accuracy. Meanwhile, the computational complexity of our MCSampEn method is hardly affected by the time series length N. We also study how the randomness of the time series affects the accuracy and computation time of the MCSampEn algorithm by applying the algorithm to the stochastic process $MIX(p)$. The results indicate that the proposed algorithm performs well for time series with less randomness.

Author Contributions: Conceptualization, Y.J.; methodology, Y.J. and W.L.; software, W.L.; validation, Y.J. and W.L.; formal analysis, Y.J. and W.L.; investigation, Y.J.; writing—original draft preparation, W.L.; writing—review and editing, Y.J. and Y.X.; visualization, W.L.; supervision, Y.J.; project administration, Y.J.; funding acquisition, Y.J. All authors have read and agreed to the published version of the manuscript.

Funding: W. Liu and Y. Jiang are supported in part by the Key Area Research and Development Program of Guangdong Province, China (No. 2021B0101190003); the Natural Science Foundation of Guangdong Province, China (No.2022A1515010831); and Science and Technology Program of Guangzhou, China (No. 201804020053). Yuesheng Xu was supported in part by US National Science Foundation under grant DMS-1912958.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used are included in the article.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In this Appendix, we provide proofs of Theorems 3–5, where Theorems 3 and 4 describe the expectations and variances of $\frac{\tilde{A}}{N_0(N_0-1)}$ and $\frac{\tilde{B}}{N_0(N_0-1)}$, and Theorem 5 presents the convergence rate of $\{-\log \frac{\tilde{B}_k}{\tilde{A}_k} : k \in \mathbb{N}\}$.

Note that the only difference in the definitions between $\frac{\tilde{A}}{N_0(N_0-1)}$ and $\frac{\tilde{B}}{N_0(N_0-1)}$ is the template length. Without loss of generality, we discuss the expectation (2) and variation (4) of $\frac{\tilde{B}}{N_0(N_0-1)}$. The Equations (1) and (3) of $\frac{\tilde{A}}{N_0(N_0-1)}$ can be obtained in a similar way.

To analyze the expectation of $\frac{\tilde{B}}{N_0(N_0-1)}$, we define the following notation. For all $j \in \mathbb{Z}_{N_0}$, we define random variable $\tilde{B}_j$ on the probability space $\{\Omega, \mathcal{F}, P\}$ by

$$\tilde{B}_j(\mathbf{s}) := \#\Big\{i \in \mathbb{Z}_{N_0} : i \neq j \text{ and } \rho(\mathbf{y}_{s_i}, \mathbf{y}_{s_j}) \leq r\Big\}, \quad \mathbf{s} \in \Omega. \tag{A1}$$

For all $j \in \mathbb{Z}_{N_0}$, the definition of $\tilde{B}_j$ indicates that $\tilde{B}_j(\mathbf{s})$ is the number of elements in $\{\mathbf{y}_{s_i} : i \in \mathbb{Z}_{N_0}\}$ that satisfy $\rho(\mathbf{y}_{s_i}, \mathbf{y}_{s_j}) \leq r$ and $i \neq j$. From the definitions of $\tilde{B}$ and $\tilde{B}_j$, we have that for all $\mathbf{s} \in \Omega$,

$$\tilde{B}(\mathbf{s}) = \frac{1}{2}\sum_{j=1}^{N_0} \tilde{B}_j(\mathbf{s}).$$

For $p, q, l \in \mathbb{N}$, we say random variable V follows the hypergeometric distribution $H(p, q, l)$ if and only if the probability of $V = k$

$$\Pr(V = k) = \begin{cases} \frac{\binom{q}{k}\binom{p-q}{l-q}}{\binom{p}{l}}, & \text{if } \max\{0, q+l-p\} \leq k \leq \min\{q, l\}, \\ 0, & \text{otherwise.} \end{cases}$$

See Section 5.3 of [29] for more details about the hypergeometric distribution. For all $l \in \mathbb{Z}_N$, let $\mathbb{B}_l := \{i \in \mathbb{Z}_N : i \neq l \text{ and } \rho(\mathbf{y}_i, \mathbf{y}_l) \leq r\}$, which is the index set of elements of Y satisfying $\rho(\mathbf{y}_i, \mathbf{y}_l) \leq r$. From the definition of B_l, we have that $B_l = \#\mathbb{B}_l$. For the purpose of analyzing the expectation of $\frac{\tilde{B}}{N_0(N_0-1)}$, we recall the expectation of the hypergeometric distribution $H(p, q, l)$ (see Theorem 5.3.2 in [29]) and prove a technical lemma as follows.

Theorem A1. *For $p, q, l \in \mathbb{N}$, the expectation of the hypergeometric distribution $H(p, q, l)$ is $\frac{ql}{p}$.*

Lemma A1. *Let $N_0 \in \mathbb{Z}_N$ with $N_0 > 1$. For any fixed $j \in \mathbb{Z}_{N_0}$ and $l \in \mathbb{Z}_N$, the conditional probability distribution of $\tilde{B}_j$ given $s_j = l$ is the hypergeometric distribution $H(N-1, B_l, N_0-1)$. Moreover, for all $j \in \mathbb{Z}_{N_0}$, the expectation of random variable $\tilde{B}_j$ is*

$$\mathrm{E}\big[\tilde{B}_j\big] = \frac{2(N_0-1)}{N(N-1)}B. \tag{A2}$$

Proof. Let $j \in \mathbb{Z}_{N_0}$ and $l \in \mathbb{Z}_N$. From the definition of $\tilde{B}_j$, we can see that for all $\mathbf{s} \in \Omega$ with $s_j = l$, $\tilde{B}_j(\mathbf{s}) \leq \min\{B_l, N_0 - 1\}$. On the other hand, since for all $\mathbf{s} \in \Omega$ with $s_j = l$,

$$\big\{i \in \mathbb{Z}_{N_0} : \rho(\mathbf{y}_{s_i}, \mathbf{y}_l) > r\big\} \subset \{i \in \mathbb{Z}_N : \rho(\mathbf{y}_i, \mathbf{y}_l) > r\},$$

from the definitions of $\tilde{B}_j$ and B_l, we have that $N_0 - \tilde{B}_j \leq N - B_l$. Thus, we can see that for all $\mathbf{s} \in \Omega$ with $s_j = l$, $\max\{0, N_0 - N + B_l\} \leq \tilde{B}_j \leq \min\{N_0 - 1, B_l\}$. This means that for $k < \max\{0, N_0 - N + B_l\}$ or $k > \min\{N_0 - 1, B_l\}$,

$$\#\{\mathbf{s} \in \Omega : \tilde{B}_j(\mathbf{s}) = k \text{ and } s_j = l\} = 0.$$

Meanwhile, it can be checked that for all $\mathbf{s} \in \Omega$ with $s_j = l$ and $\max\{0, N_0 - N + B_l\} \leq k \leq \min\{N_0 - 1, B_l\}$, $\tilde{B}_j(\mathbf{s}) = k$ if and only if vector $\mathbf{s}$ contains k components belonging to $\mathbb{B}_l$, and $N_0 - 1 - k$ components belonging to $\mathbb{Z}_N / (\mathbb{B}_l \cup \{l\})$. Note that there are $\binom{B_l}{k}$ ways of drawing k elements from set $\mathbb{B}_l$, and $\binom{N-1-B_l}{N_0-1-k}$ ways of drawing $N_0 - 1 - k$ elements from set $\mathbb{Z}_N / (\mathbb{B}_l \cup \{l\})$. Thus, by noting that each element in Ω is a permutation formed by extracting N_0 numbers from $\mathbb{Z}_N$, we have that for all $\max\{0, N_0 - N + B_l\} \leq k \leq \min\{N_0 - 1, B_l\}$,

$$\#\{\mathbf{s} \in \Omega : \tilde{B}_j(\mathbf{s}) = k \text{ and } s_j = l\} = (N_0 - 1)! \binom{B_l}{k} \binom{N - 1 - B_l}{N_0 - 1 - k}. \tag{A3}$$

Note that $\#\{\mathbf{s} \in \Omega : s_j = l\} = \frac{(N-1)!}{(N-N_0)!}$, and the elements in $\{\mathbf{s} \in \Omega : s_j = l\}$ are of equal probability. Hence, dividing the right term of (A3) by $\frac{(N-1)!}{(N-N_0)!}$, we obtain

$$P\big(\tilde{B}_j = k \mid s_j = l\big) = \begin{cases} \frac{\binom{B_l}{k}\binom{N-1-B_l}{N_0-1-B_l}}{\binom{N-1}{N_0-1}}, & \max\{0, N_0 - N + B_l\} \leq k \leq \min\{N_0 - 1, B_l\}, \\ 0, & \text{otherwise.} \end{cases}$$

This indicates that the conditional probability distribution of $\tilde{B}_j$ given $s_j = l$ is the hypergeometric distribution $H(N - 1, B_l, N_0 - 1)$ (see [29]).

Since the conditional probability distribution of $\tilde{B}_j$ given $s_j = l$ is the hypergeometric distribution $H(N - 1, B_l, N_0 - 1)$, from Theorem A1 we have for any $j \in \mathbb{Z}_{N_0}$ and $l \in \mathbb{Z}_N$, $\mathrm{E}\big[\tilde{B}_j \mid s_j = l\big] = \frac{B_l(N_0-1)}{N-1}$. Thus, by noting $\sum_{l \in \mathbb{Z}_N} B_l = 2B$ and $P\big(s_j = l\big) = \frac{1}{N}$ for all $l \in \mathbb{Z}_N$, from the law of total expectation we obtain (A2). □

The proof for Theorem 3 is shown as follows.

Proof. From the definitions of $\tilde{B}$ and $\tilde{B}_j$, we know

$$\mathrm{E}\big[\tilde{B}\big] = \frac{1}{2} \sum_{j=1}^{N_0} \mathrm{E}\big[\tilde{B}_j\big]. \tag{A4}$$

Substituting (A2) into (A4) leads to (2). □

Next we consider the variance of $\frac{\tilde{B}}{N_0(N_0-1)}$. Since $\tilde{B} = \frac{1}{2}\sum_{j \in \mathbb{Z}_{N_0}} \tilde{B}_j$, the variance of $\frac{\tilde{B}}{N_0(N_0-1)}$ can be obtained by summing the covariances $E[\tilde{B}_{j_1}, \tilde{B}_{j_2}]$, $j_1, j_2 \in \mathbb{Z}_{N_0}$. This motivates us to compute these covariances. As a preparation, we establish two auxiliary lemmas. For all $k, l \in \mathbb{Z}_N$ with $k \neq l$, we define $\mathbb{B}_{kl} := \mathbb{B}_k \cap \mathbb{B}_l$ and $B_{kl} := \#\mathbb{B}_{kl}$.

Lemma A2. *It holds that*

$$\sum_{k \in \mathbb{Z}_N} \sum_{l \in \mathbb{Z}_N \setminus \{k\}} B_{kl} = \sum_{l \in \mathbb{Z}_N} B_l^2 - 2B. \tag{A5}$$

Proof. Note that $\mathbb{B}_{kl} \bigcap \mathbb{B}_{k'l'}$ is not necessarily empty for $(k, l) \neq (k', l')$. For $\mathbb{B}_{kl}$, we define new sets Π_{kl} so that they mutually disjoint and have the same cardinality as $\mathbb{B}_{kl}$. In this way, the formula (A5) will be proved by establishing a set identity and counting their cardinality. To this end, we define $\Pi_{kl} := \{(p, k, l) : p \in \mathbb{B}_{kl}\}$, for each $k, l \in \mathbb{Z}_N$ with $k \neq l$, and $\Pi_p := \big\{(p, k, l) : k, l \in \mathbb{B}_p \text{ with } k \neq l\big\}$, for each $p \in \mathbb{Z}_N$. From the definition of Π_{kl}, we have that $\#\Pi_{kl} = B_{kl}$ and $\Pi_{kl} \bigcap \Pi_{k'l'} = \emptyset$ if $(k, l) \neq (k', l')$. Thus,

$$\sum_{k \in \mathbb{Z}_N} \sum_{l \in \mathbb{Z}_N \setminus \{k\}} B_{kl} = \#\left(\bigcup_{k \in \mathbb{Z}_N} \bigcup_{l \in \mathbb{Z}_N \setminus \{k\}} \Pi_{kl} \right). \tag{A6}$$

Likewise, the definition of Π_p ensures that $\#\Pi_p = B_p(B_p-1)$ and $\Pi_p \bigcap \Pi_{p'} = \emptyset$ if $p \neq p'$. Thus, by noting that $2B = \sum_{p\in\mathbb{Z}_N} B_p$,

$$\sum_{p\in\mathbb{Z}_N} B_p^2 - 2B = \sum_{p\in\mathbb{Z}_N} B_p(B_p-1) = \#\left(\bigcup_{p\in\mathbb{Z}_N} \Pi_p\right). \tag{A7}$$

Combining Equations (A6) and (A7), we see that it suffices to prove

$$\bigcup_{k\in\mathbb{Z}_N} \bigcup_{l\in\mathbb{Z}_N\setminus\{k\}} \Pi_{kl} = \bigcup_{p\in\mathbb{Z}_N} \Pi_p. \tag{A8}$$

For all $k,l \in \mathbb{Z}_N$ with $k \neq l$, and $(p,k,l) \in \Pi_{kl}$, the definitions of Π_{kl} and $\mathbb{B}_{kl}$ ensure

$$p \neq k,\ p \neq l,\ \rho(\mathbf{y}_p, \mathbf{y}_k) \leq r \ \text{ and } \rho(\mathbf{y}_p, \mathbf{y}_l) \leq r. \tag{A9}$$

In other words, there are $k \in \mathbb{B}_p$, $l \in \mathbb{B}_p$ and $k \neq l$. Thus, for all $k,l \in \mathbb{Z}_N$ with $k \neq l$, and $(p,k,l) \in \Pi_{kl}$, there has $(p,k,l) \in \Pi_p$. Thus, we obtain

$$\bigcup_{k\in\mathbb{Z}_N} \bigcup_{l\in\mathbb{Z}_N\setminus\{k\}} \Pi_{kl} \subset \bigcup_{p\in\mathbb{Z}_N} \Pi_p. \tag{A10}$$

On the other hand, for all $p \in \mathbb{Z}_N$ and $(p,k,l) \in \Pi_p$, we know (A9) holds and $k \neq l$ from the definitions of Π_p and $\mathbb{B}_p$. This means that $(p,k,l) \in \Pi_{kl}$ and $k \neq l$. Hence, we obtain that

$$\bigcup_{p\in\mathbb{Z}_N} \Pi_p \subset \bigcup_{k\in\mathbb{Z}_N} \bigcup_{l\in\mathbb{Z}_N\setminus\{k\}} \Pi_{kl}. \tag{A11}$$

From (A10) and (A11) we obtain (A8), which leads to the desired result (A5). □

For $i,j \in \mathbb{Z}_{N_0}$ with $i \neq j$, we define random variable Z_{ij} on the probability space $\{\Omega, \mathcal{F}, P\}$ by

$$Z_{ij}(\mathbf{s}) := \begin{cases} 1, & \text{if } \rho(\mathbf{y}_{s_i}, \mathbf{y}_{s_j}) \leq r, \\ 0, & \text{if } \rho(\mathbf{y}_{s_i}, \mathbf{y}_{s_j}) > r, \end{cases} \qquad \mathbf{s} \in \Omega. \tag{A12}$$

From the definition of $\tilde{B}_j$, we can see that $\tilde{B}_j = \sum_{i\in\mathbb{Z}_{N_0}\setminus\{j\}} Z_{ij}$. Thus, in order to compute the covariance $E[\tilde{B}_{j_1}\tilde{B}_{j_2}]$, we next show the values of $P(Z_{i_1j_1}=1, Z_{i_2j_2}=1)$ for $j_1,j_2 \in \mathbb{Z}_{N_0}$ and $i_1,i_2 \in \mathbb{Z}_{N_0}\setminus\{j_1,j_2\}$ with $i_1 \neq i_2$.

Lemma A3. *It holds that for $j_1,j_2 \in \mathbb{Z}_{N_0}$ with $j_1 \neq j_2$, and $i_1,i_2 \in \mathbb{Z}_{N_0}\setminus\{j_1,j_2\}$ with $i_1 \neq i_2$,*

$$P(Z_{i_1j_1}=1, Z_{i_2j_2}=1) = \frac{4\left(B^2+B-\sum_{l\in\mathbb{Z}_N} B_l^2\right)}{N(N-1)(N-2)(N-3)}. \tag{A13}$$

Moreover, for all $j \in \mathbb{Z}_{N_0}$ and $i,i' \in \mathbb{Z}_{N_0}\setminus\{j\}$ with $i \neq i'$, it holds that

$$P(Z_{ij}=1, Z_{i'j}=1) = \frac{\sum_{l\in\mathbb{Z}_N} B_l^2 - 2B}{N(N-1)(N-2)}. \tag{A14}$$

Proof. We first prove (A13). Let

$$\mathbb{L}_{N_0} := \{(i_1,j_1,i_2,j_2) : j_1,j_2 \in \mathbb{Z}_{N_0} \text{ with } j_1 \neq j_2, \text{ and } i_1,i_2 \in \mathbb{Z}_{N_0}\setminus\{j_1,j_2\} \text{ with } i_1 \neq i_2\}$$

and for all $(i_1,j_1,i_2,j_2) \in \mathbb{L}_{N_0}$, we define

$$\Omega_{i_1j_1,i_2j_2} := \{\mathbf{s} \in \Omega : Z_{i_1j_1}(\mathbf{s}) = 1, \text{ and } Z_{i_2j_2}(\mathbf{s}) = 1\}.$$

We prove (A13) by counting the cardinality of $\Omega_{i_1j_1,i_2j_2}$. To this end, we identify $\Omega_{i_1j_1,i_2j_2}$ as the union of disjoint subsets of $\Omega_{i_1j_1,i_2j_2}$. From the definition of $Z_{i_1j_1}$ and $Z_{i_2j_2}$, we know for all $(i_1,j_1,i_2,j_2) \in \mathbb{L}_{N_0}$ and $\mathbf{s} \in \Omega_{i_1j_1,i_2j_2}$ that $s_{i_1} \in \mathbb{B}_{s_{j_1}}$ and $s_{i_2} \in \mathbb{B}_{s_{j_2}}$. At the same time, note that for $(i_1,j_1,i_2,j_2) \in \mathbb{L}_{N_0}$ and $\mathbf{s} \in \Omega_{i_1j_1,i_2j_2}$, the numbers in set $\{s_{j_1}, s_{j_2}, s_{i_1}, s_{i_2}\}$ are distinct. Thus, for all $(i_1,j_1,i_2,j_2) \in \mathbb{L}_{N_0}$ and $\mathbf{s} \in \Omega_{i_1j_1,i_2j_2}$, it holds that $s_{j_1} \neq s_{j_2}$, $s_{i_1} \in \mathbb{B}_{s_{j_1}} \setminus \{s_{j_2}\}$, and $s_{i_2} \in \mathbb{B}_{s_{j_2}} \setminus \{s_{i_1}, s_{j_1}\}$. Namely,

$$\Omega_{i_1j_1,i_2j_2} \subset \{\mathbf{s} \in \Omega : s_{j_1} \neq s_{j_2}, s_{i_1} \in \mathbb{B}_{s_{j_1}} \setminus \{s_{j_2}\}, \text{ and } s_{i_2} \in \mathbb{B}_{s_{j_2}} \setminus \{s_{i_1}, s_{j_1}\}\}.$$

On the other hand, it is easy to check that

$$\Omega_{i_1j_1,i_2j_2} \supset \{\mathbf{s} \in \Omega : s_{j_1} \neq s_{j_2}, s_{i_1} \in \mathbb{B}_{s_{j_1}} \setminus \{s_{j_2}\}, \text{ and } s_{i_2} \in \mathbb{B}_{s_{j_2}} \setminus \{s_{i_1}, s_{j_1}\}\}.$$

Thus, for all $(i_1,j_1,i_2,j_2) \in \mathbb{L}_{N_0}$, $\Omega_{i_1j_1,i_2j_2}$ can be rewritten as

$$\Omega_{i_1j_1,i_2j_2} = \{\mathbf{s} \in \Omega : s_{j_1} \neq s_{j_2}, s_{i_1} \in \mathbb{B}_{s_{j_1}} \setminus \{s_{j_2}\}, \text{ and } s_{i_2} \in \mathbb{B}_{s_{j_2}} \setminus \{s_{i_1}, s_{j_1}\}\}.$$

For $k \neq l$, we define $\Omega^{k,l}_{i_1j_1,i_2j_2} := \{\mathbf{s} \in \Omega_{i_1j_1,i_2j_2} : s_{j_1} = k, s_{j_2} = l\}$. Then, we can rewrite $\Omega_{i_1j_1,i_2j_2}$ as

$$\Omega_{i_1j_1,i_2j_2} = \bigcup_{k\in\mathbb{Z}_N} \bigcup_{l\in\mathbb{Z}_N\setminus\{k\}} \Omega^{k,l}_{i_1j_1,i_2j_2}. \tag{A15}$$

Since $\Omega^{k,l}_{i_1j_1,i_2j_2} \cap \Omega^{k',l'}_{i_1j_1,i_2j_2} = \emptyset$ if $(k,l) \neq (k',l')$, from (A15) we can see that

$$\#\Omega_{i_1j_1,i_2j_2} = \sum_{k\in\mathbb{Z}_N} \sum_{l\in\mathbb{Z}_N\setminus\{k\}} \#\Omega^{k,l}_{i_1j_1,i_2j_2}. \tag{A16}$$

Note that for all $k \in \mathbb{Z}_N$ and $l \in \mathbb{Z}_N \setminus \{k\}$,

$$\begin{aligned}\Omega^{k,l}_{i_1j_1,i_2j_2} = \; & \{\mathbf{s} \in \Omega : s_{j_1} = k, s_{j_2} = l, s_{i_1} \in \mathbb{B}_k \setminus (\mathbb{B}_{kl} \cup \{l\}) \text{ and } s_{i_2} \in \mathbb{B}_l \setminus \{k\}\} \\ & \cup \{\mathbf{s} \in \Omega : s_{j_1} = k, s_{j_2} = l, s_{i_1} \in \mathbb{B}_{kl} \text{ and } s_{i_2} \in \mathbb{B}_l \setminus \{s_{i_1}, k\}\},\end{aligned}$$

and the two sets on the right-hand side of the above equation are disjoint. Thus, it holds that for all $k \in \mathbb{Z}_N$ and $l \in \mathbb{Z}_N \setminus \{k\}$,

$$\#\Omega^{k,l}_{i_1j_1,i_2j_2} = \frac{(N-4)!}{(N-N_0)!}((B_k - B_{kl} - Z_{kl})(B_l - Z_{kl}) + B_{kl}(B_l - Z_{kl} - 1)). \tag{A17}$$

Substituting (A17) into (A16) leads to

$$\#\Omega_{i_1j_1,i_2j_2} = \frac{(N-4)!}{(N-N_0)!} \sum_{k\in\mathbb{Z}_N} \sum_{l\in\mathbb{Z}_N\setminus\{k\}} ((B_k - B_{kl} - Z_{kl})(B_l - Z_{kl}) + B_{kl}(B_l - Z_{kl} - 1)).$$

By direct computation with noting $Z^2_{kl} = Z_{kl}$, we obtain from the equation above that

$$\#\Omega_{i_1j_1,i_2j_2} = \frac{(N-4)!}{(N-N_0)!} \sum_{k\in\mathbb{Z}_N} \sum_{l\in\mathbb{Z}_N\setminus\{k\}} (B_kB_l - B_kZ_{kl} - B_lZ_{kl} - B_{kl} + Z_{kl}). \tag{A18}$$

Note that $\sum_{k\in\mathbb{Z}_N\setminus\{l\}} Z_{kl} = B_l$ and $\sum_{l\in\mathbb{Z}_N\setminus\{k\}} B_l = 2B - B_k$. We then have that

$$\sum_{k\in\mathbb{Z}_N} \sum_{l\in\mathbb{Z}_N\setminus\{k\}} B_lZ_{kl} = \sum_{l\in\mathbb{Z}_N} \sum_{k\in\mathbb{Z}_N\setminus\{l\}} B_lZ_{kl} = \sum_{l\in\mathbb{Z}_N} B_l \left(\sum_{k\in\mathbb{Z}_N\setminus\{l\}} Z_{kl} \right) = \sum_{l\in\mathbb{Z}_N} B_l^2,$$

$$\sum_{k\in\mathbb{Z}_N}\sum_{l\in\mathbb{Z}_N\setminus\{k\}} B_kB_l = \sum_{k\in\mathbb{Z}_N} B_k(2B-B_k) = 4B^2 - \sum_{l\in\mathbb{Z}_N} B_l^2,$$

and

$$\sum_{k\in\mathbb{Z}_N}\sum_{l\in\mathbb{Z}_N\setminus\{k\}} Z_{kl} = 2B.$$

Substituting (A5) and the above equations into (A18), we obtain that

$$\#\Omega_{i_1j_1,i_2j_2} = \frac{4(N-4)!}{(N-N_0)!}\left(B^2+B-\sum_{l\in\mathbb{Z}_N} B_l^2\right). \tag{A19}$$

By noting that $\#\Omega = \frac{N!}{(N-N_0)!}$ and $P(Z_{i_1j_1}=1, Z_{i_2j_2}=1) = \frac{\#\Omega_{i_1j_1,i_2j_2}}{\#\Omega}$, we obtain (A13) from (A19).

We now turn to prove (A14). Let $j\in\mathbb{Z}_{N_0}$ and $i,i'\in\mathbb{Z}_{N_0}\setminus\{j\}$ with $i\neq i'$. Note that

$$\begin{aligned}\#\{\mathbf{s}\in\Omega : Z_{ij}(\mathbf{s}) = Z_{i'j}(\mathbf{s}) = 1\} &= \sum_{l\in\mathbb{Z}_N} \#\{\mathbf{s}\in\Omega : s_j = l, s_i\in\mathbb{B}_l \text{ and } s_{i'}\in\mathbb{B}_l\setminus\{s_i\}\}\\ &= \frac{(N-3)!}{(N-N_0)!}\sum_{l\in\mathbb{Z}_N} B_l(B_l-1).\end{aligned}$$

Thus, it holds that

$$P(Z_{ij}=1, Z_{i'j}=1) = \frac{\sum_{l\in\mathbb{Z}_N} B_l(B_l-1)}{N(N-1)(N-2)}. \tag{A20}$$

Since $\sum_{l\in\mathbb{Z}_N} B_l = 2B$, from (A20) we obtain (A14). □

With the help of Lemma A3, we can calculate $\mathrm{E}\left[\tilde{B}_{j_1}\tilde{B}_{j_2}\right]$ in the following lemma.

Lemma A4. *If $N_0\in\mathbb{Z}_N$ with $N_0>3$, then for all $j_1,j_2\in\mathbb{Z}_{N_0}$ with $j_1\neq j_2$,*

$$\begin{aligned}\mathrm{E}\left[\tilde{B}_{j_1}\tilde{B}_{j_2}\right] = {}& \frac{4(N_0-2)(N_0-3)}{N(N-1)(N-2)(N-3)}\left(B^2+B-\sum_{l\in\mathbb{Z}_N} B_l^2\right)\\ &+\frac{3(N_0-2)\left(\sum_{l\in\mathbb{Z}_N} B_l^2-2B\right)}{N(N-1)(N-2)} + \frac{2B}{N(N-1)},\end{aligned} \tag{A21}$$

and for all $j\in\mathbb{Z}_{N_0}$,

$$\mathrm{E}\left[\tilde{B}_j^2\right] = \frac{2(N_0-1)B}{N(N-1)} + \frac{(N_0-1)(N_0-2)}{N(N-1)(N-2)}\left(\sum_{l\in\mathbb{Z}_N} B_l^2 - 2B\right). \tag{A22}$$

Proof. We first prove (A21). Let $j_1,j_2\in\mathbb{Z}_{N_0}$ with $j_1\neq j_2$. From the decomposition $\tilde{B}_j = \sum_{i\in\mathbb{Z}_{N_0}\setminus\{j\}} Z_{ij}$, we obtain for all $j_1,j_2\in\mathbb{Z}_{N_0}$ with $j_1\neq j_2$ that

$$\mathrm{E}\left[\tilde{B}_{j_1}\tilde{B}_{j_2}\right] = \sum_{i_1\in\mathbb{Z}_{N_0}\setminus\{j_1\}}\sum_{i_2\in\mathbb{Z}_{N_0}\setminus\{j_2\}} \mathrm{E}\left[Z_{i_1j_1}Z_{i_2j_2}\right].$$

We further rewrite the right-hand side of the above equation to obtain

$$\begin{aligned}\mathrm{E}\left[\tilde{B}_{j_1}\tilde{B}_{j_2}\right] = {}& \sum_{i_1\in\mathbb{Z}_{N_0}\setminus\{j_1,j_2\}}\sum_{i_2\in\mathbb{Z}_{N_0}\setminus\{j_1,j_2,i_1\}} \mathrm{E}\left[Z_{i_1j_1}Z_{i_2j_2}\right]\\ &+\sum_{i_1\in\mathbb{Z}_{N_0}\setminus\{j_1,j_2\}} \mathrm{E}\left[Z_{i_1j_1}Z_{i_1j_2}\right] + \sum_{i_1\in\mathbb{Z}_{N_0}\setminus\{j_1,j_2\}} \mathrm{E}\left[Z_{i_1j_1}Z_{j_1j_2}\right]\\ &+\sum_{i_2\in\mathbb{Z}_{N_0}\setminus\{j_1,j_2\}} \mathrm{E}\left[Z_{j_2j_1}Z_{i_2j_2}\right] + \mathrm{E}\left[Z_{j_2j_1}Z_{j_1j_2}\right].\end{aligned} \tag{A23}$$

We next compute the terms on the right hand side of (A23) one by one. Since for all $j, j' \in \mathbb{Z}_{N_0}$, $i \in \mathbb{Z}_{N_0} \setminus \{j\}$ and $i' \in \mathbb{Z}_{N_0} \setminus \{j'\}$,

$$\mathrm{E}\left[Z_{ij}Z_{i'j'}\right] = P(Z_{ij} = 1, Z_{i'j'} = 1),$$

from Equation (A13) of Lemma A3, we know the first term in the right-hand side of (A23) satisfies

$$\sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j_1, j_2\}} \sum_{i_2 \in \mathbb{Z}_{N_0} \setminus \{j_1, j_2, i_1\}} \mathrm{E}\left[Z_{i_1 j_1} Z_{i_2 j_2}\right] = \frac{4\left(B^2 + B - \sum_{l \in \mathbb{Z}_N} B_l^2\right)}{N(N-1)(N-2)(N-3)} (N_0 - 2)(N_0 - 3). \tag{A24}$$

Likewise, by noting that $Z_{ij} = Z_{ji}$, from Equation (A14) of Lemma A3, we obtain the second, third, and fourth terms on the right-hand side of (A23),

$$\begin{aligned} \sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j_1, j_2\}} \mathrm{E}\left[Z_{i_1 j_1} Z_{i_1 j_2}\right] &= \sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j_1, j_2\}} \mathrm{E}\left[Z_{i_1 j_1} Z_{j_1 j_2}\right] = \sum_{i_2 \in \mathbb{Z}_{N_0} \setminus \{j_1, j_2\}} \mathrm{E}\left[Z_{j_2 j_1} Z_{i_2 j_2}\right] \\ &= (N_0 - 2) \frac{\sum_{l \in \mathbb{Z}_N} B_l^2 - 2B}{N(N-1)(N-2)}. \end{aligned} \tag{A25}$$

Note that for all $i, j \in \mathbb{Z}_{N_0}$ with $i \neq j$, it holds that $Z_{ij} = Z_{ji}$ and $Z_{ij}^2 = Z_{ij}$. Thus, the last term on the right-hand side of (A23) satisfies

$$\mathrm{E}\left[Z_{j_2 j_1} Z_{j_1 j_2}\right] = \frac{2B}{N(N-1)}. \tag{A26}$$

Substituting (A24), (A25), and (A26) into (A23) leads to (A21).

It remains to prove (A22). Since for all $j \in \mathbb{Z}_{N_0}$, $\tilde{B}_j = \sum_{i \in \mathbb{Z}_{N_0} \setminus \{j\}} Z_{ij} = \sum_{i \in \mathbb{Z}_{N_0} \setminus \{j\}} Z_{ij}^2$, there has

$$\begin{aligned} \mathrm{E}\left[\tilde{B}_j^2\right] &= \mathrm{E}\left[\sum_{i \in \mathbb{Z}_{N_0} \setminus \{j\}} Z_{ij}^2 + \sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j\}} \sum_{i_2 \in \mathbb{Z}_{N_0} \setminus \{j, i_1\}} Z_{i_1 j} Z_{i_2 j}\right], \\ &= \mathrm{E}\left[\tilde{B}_j + \sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j\}} \sum_{i_2 \in \mathbb{Z}_{N_0} \setminus \{j, i_1\}} Z_{i_1 j} Z_{i_2 j}\right] \\ &= \mathrm{E}\left[\tilde{B}_j\right] + \sum_{i_1 \in \mathbb{Z}_{N_0} \setminus \{j\}} \sum_{i_2 \in \mathbb{Z}_{N_0} \setminus \{j, i_1\}} \mathrm{E}\left[Z_{i_1 j} Z_{i_2 j}\right]. \end{aligned} \tag{A27}$$

Note that for all $j \in \mathbb{Z}_{N_0}$, $i_1 \in \mathbb{Z}_{N_0} \setminus \{j\}$ and $i_2 \in \mathbb{Z}_{N_0} \setminus \{j, i_1\}$,

$$\mathrm{E}\left[Z_{i_1 j} Z_{i_2 j}\right] = P(Z_{i_1 j} = 1, Z_{i_2 j} = 1).$$

Thus, substituting (A2) and (A14) into (A27), we obtain (A22). □

Now, we are ready to discuss the variance of $\frac{\tilde{B}}{N_0(N_0-1)}$.

The proof for Theorem 4 is shown as follows.

Proof. To prove this theorem, we compute $\mathrm{E}[\tilde{B}^2]$. Noting that $\tilde{B} = \frac{1}{2}\sum_{j=1}^{N_0} \tilde{B}_j$, we have

$$\mathrm{E}\left[\tilde{B}^2\right] = \frac{1}{4}\left(\sum_{j \in \mathbb{Z}_{N_0}} \mathrm{E}\left[\tilde{B}_j^2\right] + \sum_{j_1 \in \mathbb{Z}_{N_0}} \sum_{j_2 \in \mathbb{Z}_{N_0} \setminus \{j_1\}} \mathrm{E}\left[\tilde{B}_{j_1} \tilde{B}_{j_2}\right]\right). \tag{A28}$$

Substituting (A21) and (A22) into (A28) leads to

$$\begin{aligned}\mathrm{E}\left[\tilde{B}^2\right] = &\frac{N_0(N_0-1)}{N(N-1)}B + \frac{N_0(N_0-1)(N_0-2)}{N(N-1)(N-2)}\left(\sum_{p=1}^{N} B_p^2 - 2B\right)\\ &+\frac{N_0(N_0-1)(N_0-2)(N_0-3)}{N(N-1)(N-2)(N-3)}\left(B^2 - \sum_{p=1}^{N} B_p^2 + B\right).\end{aligned} \tag{A29}$$

Since

$$\mathrm{Var}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right] = \mathrm{E}\left[\left(\frac{\tilde{B}}{N_0(N_0-1)}\right)^2\right] - \left(\mathrm{E}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right]\right)^2,$$

by conducting some computation, from (A29) and the definition of C_{N_0} (5), we obtain (4).

We next estimate C_{N_0}. It can be checked that

$$\begin{aligned}C_{N_0} = &\frac{(N_0-2)\left(\sum_{l\in\mathbb{Z}_N} B_l^2\right)}{(N_0-1)(N-1)N(N-2)}\left(1-\frac{N_0-3}{N-3}\right)\\ &+\frac{B}{(N_0-1)N(N-1)}\left(1-2\frac{N_0-2}{N-2}+\frac{(N_0-2)(N_0-3)}{(N-2)(N-3)}\right)\\ &+\frac{B^2}{N(N-1)(N_0-1)}\left(\frac{(N_0-2)(N_0-3)}{(N-2)(N-3)}-\frac{N_0(N_0-1)}{N(N-1)}\right).\end{aligned} \tag{A30}$$

By noting $\frac{\sum_{l\in\mathbb{Z}_N} B_l^2}{(N-1)^2N} \le 1$, $\frac{(N_0-2)(N-1)}{(N_0-1)(N-2)} \le 1$ and $0 \le 1-\frac{N_0-3}{N-3} \le 1$, we know the first term in (A30) is not greater than 1. Since $\frac{B}{N(N-1)} < \frac{1}{2}$ and $1-2\frac{N_0-2}{N-2}+\frac{(N_0-2)(N_0-3)}{(N-2)(N-3)} \le \left(1-\frac{N_0-2}{N-2}\right)^2 < 1$, we have that the second term in (A30) is not greater than $\frac{1}{N_0-1}$. Note that $\frac{(N_0-2)(N_0-3)}{(N-2)(N-3)} - \frac{N_0(N_0-1)}{N(N-1)} \le 0$. Thus, we know the third term in (A30) is not positive. Therefore, we conclude that $C_{N_0} \le 1+\frac{1}{2(N_0-1)}$. □

To analyze this almost sure convergence rate of $\left\{-\log\frac{\bar{B}_k}{\bar{A}_k} : k\in\mathbb{N}\right\}$, we require Theorem 2 of [18], which is recalled as follows.

Theorem A2. *Let $\{V_i : i\in\mathbb{N}\}\cup\{V\}$ be a sequence of independent and identically distributed random variables in probability space $\{\Omega,\mathcal{F},P\}$ with expectation μ, $\sigma := \mathrm{Var}[V_i]$ and $\tau := \mathrm{E}[|V_i-\mu|]$. If $\sigma<+\infty$ and $\tau<+\infty$, then for all $0<\epsilon\le 1$ and $\beta>1$, there are constants D_β and $\tilde{D}_\beta$ (depending only on β) such that for all $i > n_{\epsilon,\beta}$,*

$$P\left(\sup_{k\ge i}\left|\frac{1}{k}\sum_{l=1}^{k} V_l - \mu\right| > \tau\epsilon\right) \le \frac{72\sigma}{\tau^2\epsilon^2 i}\left(D_\beta + \tilde{D}_\beta(\log i)^{\beta-1}\right),$$

where $n_{\epsilon,\beta}$ is defined by (6).

Combining Theorems 3, 4, and A2 leads to the almost sure convergence of $\frac{\bar{B}_{N_1}}{N_0(N_0-1)}$ and $\frac{\bar{A}_{N_1}}{N_0(N_0-1)}$ in the next lemma.

Lemma A5. *Let $\beta>1$ and $N_0\in\mathbb{Z}_N$ with $N_0>3$. Then, there are constants D_β and $\tilde{D}_\beta$ (depending only on β) such that for all $1\ge\epsilon>0$ and $N_1 > n_{\epsilon,\beta}$,*

$$P\left(\sup_{k>N_1}\left|\frac{\bar{A}_k}{N_0(N_0-1)} - \frac{A}{N(N-1)}\right| > \tau_A\epsilon\right) \le \frac{72C_{N_0}}{\tau_A^2\epsilon^2 N_0 N_1}\left(D_\beta + \tilde{D}_\beta(\log N_1)^{\beta-1}\right), \tag{A31}$$

and

$$P\left(\sup_{k>N_1}\left|\frac{\bar{B}_k}{N_0(N_0-1)}-\frac{B}{N(N-1)}\right|>\tau_B\epsilon\right)\leq\frac{72C_{N_0}}{\tau_B^2\epsilon^2N_0N_1}\left(D_\beta+\tilde{D}_\beta(\log N_1)^{\beta-1}\right), \quad \text{(A32)}$$

where $n_{\epsilon,\beta}$ *is defined by* (6).

Proof. From Theorems 3 and 4, we know that $\mathrm{Var}\left[\frac{\tilde{A}}{N_0(N_0-1)}\right]=\mathrm{Var}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right]=\frac{C_{N_0}}{N_0}<+\infty$, $\mathrm{E}\left[\frac{\tilde{A}}{N_0(N_0-1)}\right]=\frac{A}{N(N-1)}$, and $\mathrm{E}\left[\frac{\tilde{B}}{N_0(N_0-1)}\right]=\frac{B}{N(N-1)}$. Meanwhile, since $0\leq\frac{\tilde{A}}{N_0(N_0-1)}\leq 1$, $0\leq\frac{\tilde{B}}{N_0(N_0-1)}\leq 1$, $0\leq\frac{A}{N(N-1)}\leq 1$ and $0\leq\frac{B}{N(N-1)}\leq 1$, we know that $\tau_A\leq 1$ and $\tau_B\leq 1$. Thus, by Theorem A2, we obtain (A32) and (A31). □

We next consider the almost sure convergence rate of $\left\{-\log\frac{\bar{B}_k}{\bar{A}_k}:k\in\mathbb{N}\right\}$. To this end, we introduce the following lemma.

Lemma A6. *Let* $N_0\in\mathbb{Z}_N$ *with* $N_0>3$. *If* $A>0$ *and* $B>0$, *then for all* $N_1\in\mathbb{N}$ *and* $1>\epsilon>0$,

$$\begin{aligned}&P\left(\sup_{k>N_1}\left|\log\left(\frac{\bar{A}_k}{N_0(N_0-1)}\right)-\log\left(\frac{A}{N(N-1)}\right)\right|>\epsilon\right)\\ \leq\ &P\left(\sup_{k>N_1}\left|\frac{\bar{A}_k}{N_0(N_0-1)}-\frac{A}{N(N-1)}\right|>\frac{A\epsilon}{N(N-1)e}\right),\end{aligned} \quad \text{(A33)}$$

and

$$\begin{aligned}&P\left(\sup_{k>N_1}\left|\log\left(\frac{\bar{B}_k}{N_0(N_0-1)}\right)-\log\left(\frac{B}{N(N-1)}\right)\right|>\epsilon\right)\\ \leq\ &P\left(\sup_{k>N_1}\left|\frac{\bar{B}_k}{N_0(N_0-1)}-\frac{B}{N(N-1)}\right|>\frac{B\epsilon}{N(N-1)e}\right).\end{aligned} \quad \text{(A34)}$$

Proof. Note that for all $0<a,b<1$ and $0<\eta<1$, when

$$|\log a-\log b|>\eta, \quad \text{(A35)}$$

it holds that $a>be^{\eta}$, or $a<be^{-\eta}$. Hence, when (A35) holds, there is

$$a-b>b(e^{\eta}-1),\ \text{or}\ a-b<b(e^{-\eta}-1). \quad \text{(A36)}$$

By noting that $e^{\eta}-1>1-e^{-\eta}$ and $1-e^{-\eta}>e^{-1}\eta$, from (A36), we know that when (A35) holds, there has $a-b>be^{-1}\eta$, or $a-b<-be^{-1}\eta$, that is,

$$|a-b|>be^{-1}\eta. \quad \text{(A37)}$$

Note that when $a=0$, for all $0<b<1$ and $0<\eta<1$, inequality (A37) always holds. Thus, we know that for all $0\leq a<1$ and $0<b,\eta<1$, when (A35) holds, inequality (A37) holds. Then, replacing a, b, and η by $\frac{\bar{B}_k}{N_0(N_0-1)}$, $\frac{B}{N(N-1)}$ and ϵ, we know for all $N_1\in\mathbb{N}$ and $0<\epsilon<1$, when

$$\sup_{k>N_1}\left|\log\left(\frac{\bar{B}_k}{N_0(N_0-1)}\right)-\log\left(\frac{B}{N(N-1)}\right)\right|>\epsilon, \quad \text{(A38)}$$

there has

$$\sup_{k>N_1}\left|\frac{\bar{B}_k}{N_0(N_0-1)}-\frac{B}{N(N-1)}\right|>\frac{B\epsilon}{N(N-1)e}. \quad \text{(A39)}$$

Let $\mathcal{F}_1$ be the set of the events satisfying (A38), and $\mathcal{F}_2$ be the set of the events satisfying (A39). From (A38) and (A39), we know that $\mathcal{F}_1 \subset \mathcal{F}_2$. Thus, we can obtain (A34) (see Theorem 1.5.4 in [29]). Similarly, we can obtain (A33). □

Combining Lemmas A5 and A6, we obtain the almost sure convergence rate of

$$\left\{-\log\frac{\bar{B}_k}{\bar{A}_k} : k \in \mathbb{N}\right\}$$

in Theorem 5.

The proof of Theorem 5 is provided as follows.

Proof. Note that for all $N_1 \in \mathbb{N}$,

$$\begin{aligned}\sup_{k>N_1}\left|\log\frac{\bar{B}_k}{\bar{A}_k} - \log\frac{B}{A}\right| \leq & \sup_{k>N_1}\left|\log\left(\frac{\bar{B}_k}{N_0(N_0-1)}\right) - \log\left(\frac{B}{N(N-1)}\right)\right| \\ & + \sup_{k>N_1}\left|\log\left(\frac{\bar{A}_k}{N_0(N_0-1)}\right) - \log\left(\frac{A}{N(N-1)}\right)\right|.\end{aligned}$$

Thus, we know that for all $N_1 \in \mathbb{N}$ and $1 > \epsilon > 0$, if

$$\sup_{k>N_1}\left|\log\frac{\bar{B}_k}{\bar{A}_k} - \log\frac{B}{A}\right| > \max\{\tau_A, \tau_B\}\epsilon, \tag{A40}$$

then

$$\sup_{k>N_1}\left|\log\left(\frac{\bar{B}_k}{N_0(N_0-1)}\right) - \log\left(\frac{B}{N(N-1)}\right)\right| > \frac{\max\{\tau_A, \tau_B\}\epsilon}{2} \geq \frac{\tau_B\epsilon}{2}, \tag{A41}$$

or

$$\sup_{k>N_1}\left|\log\left(\frac{\bar{A}_k}{N_0(N_0-1)}\right) - \log\left(\frac{A}{N(N-1)}\right)\right| > \frac{\max\{\tau_A, \tau_B\}\epsilon}{2} \geq \frac{\tau_A\epsilon}{2}. \tag{A42}$$

Let $\mathcal{F}_1$ be the set of the events satisfying (A40), $\mathcal{F}_2$ be the set of the events satisfying (A41), and $\mathcal{F}_3$ be the set of events satisfying (A42). Then, from the above inequalities, we have $\mathcal{F}_1 \subset \mathcal{F}_2 \cup \mathcal{F}_3$. Hence, we have $P(\mathcal{F}_1) \leq P(\mathcal{F}_2) + P(\mathcal{F}_3)$ (see Theorems 1.5.4 and 1.5.7 in [29]), that is,

$$\begin{aligned}& P\left(\sup_{k>N_1}\left|\log\frac{\bar{B}_k}{\bar{A}_k} - \log\frac{B}{A}\right| > \max\{\tau_A, \tau_B\}\epsilon\right) \\ \leq\ & P\left(\sup_{k>N_1}\left|\log\left(\frac{\bar{B}_k}{N_0(N_0-1)}\right) - \log\left(\frac{B}{N(N-1)}\right)\right| > \frac{\tau_B\epsilon}{2}\right) \\ & + P\left(\sup_{k>N_1}\left|\log\left(\frac{\bar{A}_k}{N_0(N_0-1)}\right) - \log\left(\frac{A}{N(N-1)}\right)\right| > \frac{\tau_A\epsilon}{2}\right).\end{aligned}$$

Substituting (A34) and (A33) into above inequality, from Lemma A5 and the definitions of γ_A and γ_B, we obtain the desired result (7). □

References

1. Pincus, S.M. Approximate entropy as a measure of system complexity. *Proc. Natl. Acad. Sci. USA* **1991**, *88*, 2297–2301. [CrossRef] [PubMed]
2. Richman, J.S.; Moorman, J.R. Physiological time-series analysis using approximate entropy and sample entropy. *Am. J. Physiol. Heart Circ. Physiol.* **2000**, *278*, 2039–2049. [CrossRef] [PubMed]
3. Costa, M.; Goldberger, A.L.; Peng, C.-K. Multiscale entropy analysis of complex physiologic time series. *Phys. Rev. Lett.* **2002**, *89*, 068102. [CrossRef] [PubMed]

4. Jiang, Y.; Peng, C.-K.; Xu, Y. Hierarchical entropy analysis for biological signals. *J. Comp. Appl. Math.* **2011**, *236*, 728–742. [CrossRef]
5. Li, Y.; Li, G.; Yang, Y.; Liang, X.; Xu, M. A fault diagnosis scheme for planetary gearboxes using adaptive multi-scale morphology filter and modified hierarchical permutation entropy. *Mech. Syst. Signal Proc.* **2017**, *105*, 319–337. [CrossRef]
6. Yang, C.; Jia, M. Hierarchical multiscale permutation entropy-based feature extraction and fuzzy support tensor machine with pinball loss for bearing fault identification. *Mech. Syst. Signal Proc.* **2021**, *149*, 107182. [CrossRef]
7. Li, W.; Shen, X.; Li, Y. A comparative study of multiscale sample entropy and hierarchical entropy and its application in feature extraction for ship-radiated noise. *Entropy* **2019**, *21*, 793. [CrossRef]
8. Jiang, Y.; Mao, D.; Xu, Y. A fast algorithm for computing sample entropy. *Adv. Adapt. Data Anal.* **2011**, *3*, 167–186. [CrossRef]
9. Mao, D. Biological Time Series Classification via Reproducing Kernels and Sample Entropy. Ph.D. Dissertation, Syracuse University, Syracuse, NY, USA, August 2008.
10. Grassberger, P. An optimized box-assisted algorithm for fractal dimensions. *Phys. Lett. A* **1990**, *148*, 63–68. [CrossRef]
11. Theiler, J. Efficient algorithm for estimating the correlation dimension from a set of discrete points. *Phys. Rev. A Gen. Phys.* **1987**, *36*, 4456–4462. [CrossRef]
12. Manis, G. Fast computation of approximate entropy. *Comput. Meth. Prog. Biomed.* **2008**, *91*, 48–54. [CrossRef]
13. Manis, G.; Aktaruzzaman, M.; Sassi, R. Low computational cost for sample entropy. *Entropy* **2018**, *20*, 61. [CrossRef]
14. Wang, Y.H.; Chen, I.Y.; Chiueh, H.; Liang, S.F. A low-cost implementation of sample entropy in wearable embedded systems: An example of online analysis for sleep eeg. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 9312616. [CrossRef]
15. Tomčala, J. New fast ApEn and SampEn entropy algorithms implementation and their application to supercomputer power consumption. *Entropy* **2020**, *22*, 863. [CrossRef] [PubMed]
16. Shekelyan, M.; Cormode, G. Sequential Random Sampling Revisited: Hidden Shuffle Method. In Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, Virtually Held, 13–15 April 2021; pp. 3628–3636.
17. Karr, A.F. *Probability*; Springer: New York, NY, USA, 1993.
18. Luzia, N. A simple proof of the strong law of large numbers with rates. *Bull. Aust. Math. Soc.* **2018**, *97*, 513–517. [CrossRef]
19. Goldberger, A.L.; Amaral, L.A.; Glass, L.; Hausdorff, J.M.; Ivanov, P.C.; Mark, R.G.; Mietus, J.E.; Moody, G.B.; Peng, C.-K.; Stanley, H.E. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *Circulation* **2000**, *101*, 215–220. [CrossRef] [PubMed]
20. Shao, S.; McAleer, S.; Yan, R.; Baldi, P. Highly accurate machine fault diagnosis using deep transfer learning. *IEEE Trans. Ind. Inform.* **2019**, *15*, 2446–2455. [CrossRef]
21. Case Western Reserve University Bearing Data Center. Available online: https://engineering.case.edu/bearingdatacenter (accessed on 27 March 2022).
22. Royal Netherlands Meteorological Institute. Available online: https://www.knmi.nl/nederland-nu/klimatologie/uurgegevens (accessed on 27 March 2022).
23. Petrutiu, S.; Sahakian, A.V.; Swiryn, S. Abrupt changes in fibrillatory wave characteristics at the termination of paroxysmal atrial fibrillation in humans. *Europace* **2007**, *9*, 466–470. [CrossRef]
24. Jager, F.; Taddei, A.; Moody, G.B.; Emdin, M.; Antolič, G.; Dorn, R.; Smrdel, A.; Marchesi, C.; Mark, R.G. Long-term st database: a reference for the development and evaluation of automated ischaemia detectors and for the study of the dynamics of myocardial ischaemia. *Med. Biol. Eng. Comput.* **2003**, *41*, 172–182. [CrossRef]
25. Baim, D.S.; Colucci, W.S.; Monrad, E.S.; Smith, H.S.; Wright, R.F.; Lanoue, A.; Gauthier, D.F.; Ransil, B.J.; Grossman, W.; Braunwald, E. Survival of patients with severe congestive heart failure treated with oral milrinone. *J. Am. Coll. Cardiol.* **1986**, *7*, 661–670. [CrossRef]
26. Welch, J.; Ford, P.; Teplick, R.; Rubsamen, R. The massachusetts general hospital-marquette foundation hemodynamic and electrocardiographic database–comprehensive collection of critical care waveforms. *Clin. Monit.* **1991**, *7*, 96–97.
27. Shoeb, A.H. Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment. Ph. D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, September 2009.
28. Silva, L.E.V.; Filho, A.C.S.S.; Fazan, V.P.S.; Felipe, J.C.; Junior, L.O.M. Two-dimensional sample entropy: Assessing image texture through irregularity. *Biomed. Phys. Eng. Expr.* **2016**, *2*, 045002. [CrossRef]
29. DeGroot, M.H.; Schervish, M.J. *Probability and Statistics*, 4th ed.; Person Education: New York, NY, USA, 2012.