ENTROPY ESTIMATION METHODS AND HEALTH TESTS FOR CRYPTOGRAPHIC RANDOM NUMBER GENERATORS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MELİS ASLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
CRYPTOGRAPHY

SEPTEMBER 2024

Approval of the thesis:

ENTROPY ESTIMATION METHODS AND HEALTH TESTS FOR CRYPTOGRAPHIC RANDOM NUMBER GENERATORS

submitted by **MELİS ASLAN** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Cryptography Department, Middle East Technical University** by,

ABSTRACT

ENTROPY ESTIMATION METHODS AND HEALTH TESTS FOR CRYPTOGRAPHIC RANDOM NUMBER GENERATORS

Aslan, Melis
Ph.D., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Ali Doğanaksoy

September 2024, 70 pages

Random numbers play a crucial role in cryptography since the security of cryptographic protocols relies on the assumption of the availability of uniformly distributed and unpredictable random numbers to generate secret keys, passwords, initialization vectors, nonces, salt, etc. True Random Number Generators (TRNGs) extract random numbers from physical processes (such as radioactive decay, thermal noise, and atmospheric noise) that are inherently unpredictable. However, real-world random number generators sometimes fail and produce outputs with low entropy, leading to security vulnerabilities. It is commonly observed that the TRNG outputs have statistical biases and dependencies that make them unsuitable to be directly used for cryptographic purposes. There are some standards and guidelines on generating and testing random numbers that are suitable for cryptographic applications.

The National Institute of Standards and Technology (NIST) Special Publication (SP) 800-90 series provide guidelines and recommendations for generating random numbers for cryptographic applications and describes statistical randomness testing, estimating min-entropy with 10 black-box entropy estimation methods. In this thesis, we evaluate the effectiveness and limitations of the SP 800-90B methods by exploring the accuracy of these estimators using simulated random numbers with known entropy, investigating the correlation between entropy estimates, and studying the impacts of deterministic transformations on the estimators.

To understand the unpredictability of the outputs, it is important to estimate their entropy accurately. The National Institute of Standards and Technology (NIST) Special Publication (SP) 800-90B specifies ten *min-entropy* estimators ranging from simple frequency-based estimators to more advanced approaches. Each estimator has specific assumptions, making them suitable for different types of sources. The minimum of these estimates is assumed to be the min-entropy of the TRNG. We propose a new entropy estimator (estimates min-entropy and also *Shannon* entropy) called *index-value coincidence estimate* that is suitable for outputs that might include some dependencies and we also provide some experimental results that demonstrate the effectiveness of the estimator.

Additionally, TRNGs may be affected by outside conditions such as temperature, humidity, etc. Health tests are an integral part of the noise source of TRNG, defined to detect unexpected changes in the working process and dramatic changes in the amount of entropy generated by the noise source. Existing health test suites are examined, and a health test suite for cryptographic TRNGs is introduced by using random variables *weight, run, runs of length 1* and *overlapping templates*. Some suggested parameters and experimental results are given.

Keywords: Random number generators, entropy, randomness, statistical randomness tests, health tests, correlation

ÖZ

KRİPTOGRAFİK RASTGELE SAYI ÜRETEÇLERİ İÇİN ENTROPİ TAHMİN YÖNTEMLERİ VE SAĞLIK TESTLERİ

Aslan, Melis
Doktora, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Ali Doğanaksoy

Eylül 2024, 70 sayfa

Rastgele sayılar kriptografide önemli bir rol oynar, kriptografik protokollerin güvenliği gizli anahtarlar, parolalar, başlatma vektörleri, gürültü, maskeleme vektörleri gibi dizileri üretmek için düzgün dağılıma sahip ve öngörülemeyen rastgele sayıların kullanılabilirliği varsayımına dayanır. Gerçek Rastgele Sayı Üreteçleri (GRSÜ'ler), çeşitli fiziksel olgulardan (radyoaktif bozunma, termal gürültü ve atmosferik gürültü gibi) rastgele sayılar üretirler ve bu üretim doğası gereği rastgele olarak kabul edilir. fakat pratik uygulamalarda, rastgele sayı üreteçleri bazen başarısız olur ve düşük entropili çıktılar üretir, bu da güvenlik açıklarına yol açar. Genel olarak gözlemlendiği üzere, Gerçek Rastgele Sayı Üreteçleri (GRSÜ) çıktılarında istatistiksel sapma ve bağımlılıklar bulunmaktadır ve bu nedenle doğrudan kriptografik amaçlar için kullanılmaya uygun değildirler. Kriptografik uygulamalar için uygun rastgele sayı üretme ve test metotları konusunda standartlar ve kılavuzlar mevcuttur.

Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) Özel Yayını (SP) 800-90 serisi, kriptografik uygulamalar için rastgele sayı üretme konusunda kılavuzlar ve öneriler sunmakta, istatistiksel rastgelelik testleri ve 10 kara kutu entropi tahmin yöntemi tanımlamaktadır. Bu tezde, NIST SP 800-90B entropi tahmin yöntemlerinin etkinliğini ve sınırlamalarını, bilinen entropili simüle edilmiş rastgele sayılar kullanarak deneysel olarak değerlendirmekte, bu tahmin edicilerin doğruluğu, entropi tahmin-leri arasındaki korelasyon ve deterministik dönüşümlerin tahmin ediciler üzerindeki etkileri üzerine gözlemler yapılmaktadır.

Çıktıların öngörülemezliğini anlamak için entropilerini doğru bir şekilde tahmin etmek önemlidir. Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) Özel Yayını (SP) 800-90B, basit frekans tabanlı tahmin edicilerden daha gelişmiş yaklaşımlara kadar uzanan on adet minimum entropi tahmin edicisini belirlemektedir. Her tahmin edicinin belirli varsayımları vardır ve bu da onları farklı kaynak türlerini değerlendirmeye uygun hale getirir. Bu tahminlerin minimumunun, GRSÜ'nin minimum entropisi olduğu varsayılır. Bu tezde, bazı bağımlılıklar içerebilecek çıktıları değerlendirmek için uygun olduğu öngörülen *indeks-değer çakışması tahmini* adı verilen yeni bir entropi tahmin yöntemi tanımlanmaktadır ve yöntemin etkinliğini gösteren bazı deneysel sonuçlar sunulmaktadır.

Ek olarak, GRSÜ'leri çalışmaları sırasında sıcaklık, nem vb. gibi dış koşullardan etkilenebilir. Sağlık testleri, çalışma sürecindeki beklenmedik değişiklikleri ve gürültü kaynağı tarafından üretilen entropi miktarındaki dramatik değişiklikleri tespit etmek için tanımlanan GRSÜ'nin gürültü kaynağının bileşeni olarak tanımlanmaktadır. Çalışmalar kapsamında literatürde bulunan sağlık testi paketleri incelenmiş ve ağırlık, öbek , 1-uzunluğunda öbekler ve örtüşen kalıplar rastgele değişkenlerini kullanarak kriptografik GRSÜ'leri için bir sağlık testi paketi tanımlanmıştır, test paketi için önerilen parametreler ve deneysel sonuçlar sunulmaktadır.

Anahtar Kelimeler: Rastgele sayı üreteçleri, entropi, rastgelelik, istatistiksel rastgelelik testleri, sağlık testleri, korelasyon

To street cats...

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Ali Doğanaksoy, for his invaluable guidance, support, and encouragement throughout my academic journey. His expertise in cryptography and his unwavering belief in my abilities have been instrumental in shaping this dissertation. Beyond his academic mentorship, Ali Doğanaksoy has been a constant source of inspiration and support in my personal life. His guidance has not only helped me grow academically but has also shaped me into a more well-rounded individual.

I am also deeply indebted to Fatih Sulak for his academic and moral support, encouragement throughout my academic journey. His insights and experience on cryptography have significantly enhanced the quality of my work. I would also like to thank Meltem Sönmez Turan and Zülfükar Saygı for their guidance and advice.

I am grateful to Seda Odacıoğlu for her technical expertise and support in developing the software used in this research and conducting the experiments.

I would like to express my heartfelt thanks to my close friend, Pelin Çiloğlu, for her unwavering support and companionship throughout our academic journey.

Finally, I would like to express my sincere gratitude to FAME CRYPT for providing me with the opportunity to apply my academic knowledge to real-world projects. My involvement in the projects allowed me to gain valuable experience in applications of cryptography. The company's support was instrumental in my personal and professional growth. I am particularly grateful for the chance to work with such an experienced research group. The skills and knowledge I acquired at FAME CRYPT have equipped me to pursue a career in cryptography.

TABLE OF CONTENTS

ABSTRACT vii

ÖZ ix

ACKNOWLEDGMENTS xiii

TABLE OF CONTENTS xv

LIST OF TABLES xix

LIST OF FIGURES xxiii

LIST OF ABBREVIATIONS xxv

CHAPTERS

1 INTRODUCTION 1

2 PRELIMINARIES 7

2.1 Random Number Generators 7

2.2 Entropy 9

2.3 Correlation 11

3 OBSERVATIONS ON NIST SP 800-90B 13

3.0.1 Entropy Estimation Based on NIST SP 800-90B . . 13

3.1 Methodology 15

3.1.1 Entropy Estimation using Known Distributions . . 16

3.1.2 Correlation of the Entropy Estimators 16

3.1.3 Impact of Deterministic Transformations 16

3.1.4 Entropy Estimation According to IID-Assumption Tests using Known Distributions 17

3.2 Experimental Results 18

3.2.1 Accuracy of Entropy Estimators 18

3.2.2 Correlations of Estimators 23

3.2.2.1 Correlation Analysis with Dataset 1: IID-Full Entropy 23

3.2.2.2 Correlation Analysis with Dataset 2: IID-Low Entropy 25

3.2.2.3 Correlation Analysis with Dataset 3: Non-IID-Low Entropy 30

3.2.3 Impact of the Transformations 32

3.2.4 Impact of the IID-Assumption Tests 33

3.3 Discussion 37

4 ESTIMATING ENTROPY VIA INDEX-VALUE COINCIDENCE . . 39

4.1 Index-Value Coincidence Estimator 39

4.1.1 Estimating the Probability Distribution 40

4.1.2 Description of the Index-Value Coincidence Estimator 42

4.2 Experimental Results 44

4.2.1 Comparison for *min-entropy*: Index-Value vs. NIST SP 800-90B Estimators 45

4.2.2 Comparison for *Shannon-entropy*: Index-Value vs. NIST SP 800-90B Estimators 46

4.2.3 Correlation of Estimators 47

5 HEALTH TESTS FOR CRYPTOGRAPHIC RANDOM NUMBER GENERATORS 49

5.1 Details of Existing Health Test Suites 50

5.2 Proposed Health Tests 51

5.2.1 Test Description 52

5.2.2 Random Variables 54

5.2.2.1 Weight 54

5.2.2.2 Number of Total Runs 54

5.2.2.3 Number of Runs of Length-1 55

5.2.2.4 Overlapping Templates 56

5.3 Application 58

5.3.1 Experimental Parameters and Boundary Values . . 58

5.3.2 Experimental Results 59

6 CONCLUSION 63

REFERENCES 65

CURRICULUM VITAE 69

LIST OF TABLES

Table 2.1 Interpretation of Pearson r and Spearman ρ correlation coefficients . 12

Table 3.1 Test statistics of Permutation Tests of NIST SP 800-90B 14

Table 3.2 Entropy Estimators of NIST SP 800-90B 15

Table 3.3 Mean and standard deviation of entropy estimators for binary, 4-bit, and 8-bit sources with full entropy 19

Table 3.4 Mean and standard deviation of entropy estimators of datasets for biased binary, 4-bit near-uniform, and 8-bit near-uniform distributions . . 19

Table 3.5 Pearson correlation among different estimators for uniform distribution with full entropy $1\,000\,000$ 23

Table 3.6 Spearman correlation among different estimators for uniform distribution with full entropy $1\,000\,000$ 24

Table 3.7 Pearson correlation among different estimators for uniform distribution with full entropy $8\,000\,000$ 24

Table 3.8 Spearman correlation among different estimators for uniform distribution with full entropy $8\,000\,000$ 24

Table 3.9 Pearson correlation among different estimators for uniform distribution with full entropy $32\,000\,000$ 24

Table 3.10 Spearman correlation among different estimators for uniform distribution with full entropy $32\,000\,000$ 25

Table 3.11 Pearson correlation among different estimators for binary biased distribution with 0.5 entropy 27

Table 3.12 P-values of Pearson correlation among different estimators for binary biased distribution with 0.5 entropy 27

Table 3.13 Spearman correlation among different estimators for binary biased distribution with 0.5 entropy 27

Table 3.14 P-values of Spearman correlation among different estimators for binary biased distribution with 0.5 entropy 28

Table 3.15 Pearson correlation among different estimators for 4-bit biased distribution with 0.5 entropy 28

Table 3.16 P-values of Pearson correlation among different estimators for 4-bit biased distribution with 0.5 entropy 28

Table 3.17 Spearman correlation among different estimators for 4-bit biased distribution with 0.5 entropy 29

Table 3.18 P-values of Spearman correlation among different estimators for 4-bit biased distribution with 0.5 entropy 29

Table 3.19 Pearson correlation among different estimators for 8-bit biased distribution with 0.5 entropy 29

Table 3.20 P-values of Pearson correlation among different estimators for 8-bit biased distribution with 0.5 entropy 30

Table 3.21 Spearman correlation among different estimators for 8-bit biased distribution with 0.5 entropy 30

Table 3.22 P-values of Spearman correlation among different estimators for 8-bit biased distribution with 0.5 entropy 30

Table 3.23 Pearson correlation among different estimators for Non-IID binary biased distribution with low entropy 31

Table 3.24 P-values of Pearson correlation among different estimators for non-iid binary biased distribution with low entropy 31

Table 3.25 Spearman correlation among different estimators for non-iid binary biased distribution with low entropy 32

Table 3.26 P-values of Spearman correlation among different estimators for non-iid binary biased distribution with low entropy 32

Table 3.27 Pearson Correlation according to the estimation results of transformed sequences 33

Table 3.28 Spearman Correlation according to the estimation results of transformed sequences 33

Table 3.29 IID-Assumption Test Results 35

Table 3.30 Entropy Estimation Results of IID sequences 36

Table 3.31 Entropy Estimation Results of Non-IID sequences 36

Table 4.1 Example demonstration of index-value coincidences 40

Table 4.3 Example index-value coincidences values for a binary sequence . . . 43

Table 4.4 Mean and standard deviation of min-entropy estimators for uniform distributions: binary with full entropy, and 4-bit with full entropy sources 45

Table 4.5 Mean and standard deviation of min-entropy estimators for biased distributions: binary with 0.5 entropy, 4-bit with 0.5 entropy, and 8-bit with 0.5 entropy sources 45

Table 4.6 Mean and standard deviation of Shannon-entropy estimations for uniform binary distribution with full entropy, and biased binary distribution sources 47

Table 4.7 Pearson correlation among index-value coincidence and NIST SP 800-90B estimators for uniform binary distribution with full entropy . . . 47

Table 4.8 Spearman correlation among index-value coincidence and NIST SP 800-90B estimators for uniform binary distribution with full entropy . . . 48

Table 5.1 Health Tests Boundary Values for 1 million bit 58

Table 5.2 Health Tests Boundary Values for 20 000 bit 59

Table 5.3 Health tests results for simulated data sets with Parameter Set 1 . . . 60

Table 5.4 Health tests results for simulated data sets with Parameter Set 2 . . . 61

LIST OF FIGURES

Figure 2.1 True Random Number Generator Model 8

Figure 3.1 Distribution of Entropy Estimations for Binary Uniform Distribution Dataset 20

Figure 3.2 Distribution of Entropy Estimations for 4-bit Uniform Distribution Dataset 21

Figure 3.3 Distribution of Entropy Estimations for 8-bit Uniform Distribution Dataset 22

Figure 5.1 Probability Distribution of Random Variable X 52

LIST OF ABBREVIATIONS

AES	Advance Encryption Standard
BSI	The Federal Office for Information Security of Germany
CBC	Cipher Block Chaining
ISO	International Organization for Standardization
NIST	National Institute of Standards an Technology
PRNG	Pseudo-Random Number Generator
RNG	Random number Generator
TRNG	True-Random Number Generator

CHAPTER 1

INTRODUCTION

Throughout history, people have developed many solutions to ensure the security of information they share through insecure channels. Primitive examples of encrypted texts were seen in Egyptian hieroglyphs, and during the Roman period, the Caesar cipher, which was created by shifting each letter of the alphabet forward a fixed number of letters, is considered one of the first encryption methods in history. In parallel with technological advances, encryption methods have also developed and changed. In the computer age, encryption machines have given way to mathematical algorithms, and powerful and advanced cryptographic functions that provide solutions to many needs have been designed. Today, there are cryptographic algorithms designed in a structure that even quantum computers cannot break. However, the security of these algorithms, which are structurally known with all their details, depends on their secret parts: cryptographic keys, passwords, nonce, salt, masking vectors, etc. These are just ***random number sequences...***

Random numbers are widely used in almost all cryptographic protocols to generate secret keys, initialization vectors, nonce, salts, etc. The security of these protocols relies on the assumption that these numbers are generated uniformly at random and are unpredictable. Randomness of a sequence can be defined by three features; uniformity, independence and unpredictability.

Definition 1.0.1. *Let S be a sequence of length n, generated by the elements of the finite set $\mathbb{A} = \{a_1, a_2, \cdots, a_m\}$. S is accepted as a* ***random sequence*** *if the followings hold:*

- *Each element of $\mathbb{A}$ occurs in S with probability $\frac{1}{m}$.*
- *Each element of $\mathbb{A}$ is distributed in S uniformly.*
- *Each element of $\mathbb{A}$ is distributed in S independently.*

For cryptographic purposes, random numbers are generated by random number generators (RNGs); they can be classified into two types: true random number generators (TRNGs) and pseudo-random number generators (PRNGs). TRNGs generate sequences by measuring some physical phenomena that generate entropy, such as radioactive decay, atmospheric noise, thermal noise, movement of an electron, etc. On the other hand, PRNGs are deterministic algorithms and they extend seeds to long random-looking sequences. However, real-world random number generators sometimes fail to generate random sequences and produce outputs with low entropy, leading to security vulnerabilities [10, 6].

A variety of organizations have developed standards and guidelines on generating random numbers that are suitable for cryptographic applications, such as the National Institute of Standards of Technology (NIST) [3, 32, 4, 28], the International Organization for Standardization (ISO) [14, 15, 13, 16], and Bundesamt für Sicherheit in der Informationstechnik (BSI) [1, 2, 25].

Cryptographic random number generators are typically composed of multiple components, including (i) a *noise source* that extracts randomness from physical phenomena (e.g., thermal noise, mouse movements, radioactive decay, free-running oscillator) to generate a *seed*,(ii) a *conditioning component - pseudorandom number generator* (PRNG) (also known as a *deterministic random bit generator*) that extends the seed to generate a long random-looking sequence, and (iii) *health tests* that verify the ongoing functionality of the noise source. Since PRNGs are deterministic, the entropy is solely provided by the noise source, and it is important to measure the unpredictability of the noise source outputs.

Designing random number generators for cryptographic use has many challenges, including finding a robust *noise source* to extract randomness, difficulty of determining how unpredictable the outputs are (i.e., estimating its entropy), difficulty of statistically modeling the process, difficulty of measuring the effects of environmental conditions (e.g. pressure, humidity, temperature) on the source.

Various statistical randomness tests can be applied to measure the quality of the random numbers. The most commonly used statistical randomness suites are TESTU01 [21], DIEHARD [23], DIEHARDER [7], and NIST Special Publication (SP) 800-22 Rev.1 [27]. These tests may not be suitable for assessing noise source outputs, as they typically have strong biases and would fail these tests.

The unpredictability of noise source outputs is measured using *entropy*, and two commonly used measures of entropy are *Shannon entropy* and *min-entropy*. *Min-entropy* is a more conservative measure, which is based on the probability of guessing the most likely output of a randomness source.

Estimating the entropy of noise source outputs is challenging because the distribution of the output values is generally unknown. The BSI standards require stochastic modeling of the noise source to specify a family of probability distributions to estimate entropy. Since stochastic modeling may not be possible or practical due to the diversity and complexity of the random number generators, NIST standards allow using black-box statistical methods for entropy estimation. It is also challenging to construct statistical models for estimating entropy, and especially *Shannon-entropy*, it requires probability estimations for each character in the alphabet set. For these reasons, when number of statistical tests in statistical randomness test suits and entropy estimation suites are compared, it is observed that the number of statistical randomness test are much more than entropy estimators. NIST SP 800-22A A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications offers 15 different statistical tests; on the other hand, NIST SP 800-90B contains 10 entropy estimators.

NIST SP 800-90B [32] introduces designing and testing models for entropy sources. For testing process, SP 800-90B describes IID-Assumption tests: permutation tests and additional chi-square tests, and ten entropy estimators: most common value, collision, Markov, compression, t-tuple, longest repeated substring (LRS), multi most common in window prediction, lag prediction, multiple Markov Model with Counting (multiMMC) prediction, and LZ78Y. SP 800-90B defines two tracks to estimate the min-entropy of an entropy source: independent and identically distributed (IID) and non-IID. If a sequence passes IID-Assumption tests, entropy estimation is done via the most common value estimate; otherwise, the minimum of these ten estimates is used to estimate the min-entropy of the noise source outputs. As a designing component of an entropy source, this recommendation describes some requirements for health tests and provides two approved health tests.

NIST SP 800-90 series serve as a foundational standard for building trust in the randomness used within cryptographic systems in Cryptographic Algorithm Validation Program (CAVP) and the Cryptographic Module Validation Program (CMVP). The role of NIST SP 800-90B in CAVP and CMVP underscores its significance in ensuring the security of cryptographic modules. NIST often incorporates feedback from the community to refine its standards and guidelines, NIST SP 800-90B is currently under revision. Developments in the area of random number generation and entropy source verification are ongoing.

In literature, there are many studies evaluate accuracy of NIST SP 800-90B and to increase the accuracy introduce new entropy estimation methods for cryptographic RNGs. Zhu et al. [35] showed that the collision and compression estimates provide significant underestimates and proposed a new estimator that achieves better accuracy for min-entropy. Kim et al. [17] also showed that the compression estimate underestimates min-entropy and proposed two kinds of min-entropy estimators to improve computational complexity and estimation accuracy by leveraging two variations of Maurer's test. Hill [11] demonstrated that the collision and compression estimators incorrectly use the central limit theorem. Hill [11] also claimed that the Markov estimator should not be directly compared to other estimators since it does not use confidence intervals during estimation. Ma et al. [22] introduced a stochastic model for estimating the entropy of ADC Sampling-Based True Random Number Generators, Li et al. [20] introduced a model for estimating min-entropy based on Pruning and Quantized Deep Neural Network. Woo et al. [34] generalized LRS Estimator for min-entropy estimation, and according to their experimental results generalized LRS estimator improved the estimation accuracy significantly.

Additionally, for such test suites containing several number of statistical tests, it is important to investigate the statistical relations and independence of each individual test for efficiency of the suite. For statistical randomness tests, there are some studies investigating the relations and independence of individual tests. Turan et al. [33] provided a correlation and sensitivity analysis of statistical randomness tests, Sulak et al. [31] analyzed independence of statistical randomness tests included in the NIST SP 800-22A and Doğanaksoy et al. [8] analyzed mutual correlation of NIST 800-22A statistical randomness tests and compared of their sensitivities on transformed sequences.

In this thesis, we focus on black-box statistical methods for entropy estimation. The accuracy of the estimation process of NIST SP 800-90B [32] is analyzed with numerous statistical experiments. This study also evaluates effectiveness, and limitations of the SP 800-90B estimators using simulated random numbers with known entropy, investigates the correlation between entropy estimates, and studies the impacts of deterministic transformations on the estimators. We propose a new entropy estimator called *index-value coincidence estimate*. This estimator is used to estimate both min-entropy and *Shannon entropy* as the technique first estimates the probability distribution. Moreover, we investigate the details of existing health tests for TRNGs. We construct a statistical model by using distributions of random variables *weight, run, runs of length 1* and *overlapping templates*. We introduce a health test suite for cryptographic random number generators.

Contributions of this thesis. In this thesis, we mainly focus on evaluation of noise source components of TRNGs with entropy estimation methods and health tests. NIST SP 800-90B is the most commonly used standard for this purpose; in literature, there are some studies about techniques of SP 800-90B. However, this thesis gives a more comprehensive analysis of this standard consisting of the accuracy of estimators, the impact of IID-assumption tests, the correlation analysis of estimators, and the impact of deterministic transformations on entropy estimators. To analyze accuracy of entropy estimators, different data generation methods are designed and simulated datasets are generated with known entropy, with these datasets we analysed the evaluation power of the test suite and each individual estimator. Similarly, for IID-assumption tests, we design some biased generation methods to observe the accuracy of tests and their impact on entropy estimation. In literature, this is the first study that gives a comprehensive analysis of the mutual correlation of NIST SP 800-

90B estimators by using two correlation metrics, Pearson correlation and Spearman correlation. Moreover, some deterministic transformations are used to analyze the changes in the entropy estimation results and promote similar studies to consider the impacts of commonly used conditioning or post-processing functions. The results of this part of the thesis may help to improve the accuracy of NIST's entropy estimation strategy and help users of NIST SP 800-90B to understand and analyze the results of the test suite.

The second contribution of this thesis is that a new entropy estimation method: index-value coincidence has been introduced. This estimator is a black-box statistical entropy estimation method, gives accurate estimation results for simulated datasets and experiments show that it can be employed synchronously with NIST estimators to estimate min-entropy. Moreover, in the literature, there is no statistical-based estimator for estimating the Shannon entropy of noise sources. Index-value coincidence estimates Shannon entropy. For binary outputs, Shannon entropy estimates of NIST SP 800-90B estimators and index-value coincidence are compared, and experimental results show that index-value coincidence estimate gives more accurate estimations.

As the third contribution of this thesis, we introduce a mathematical model of health test suites for TRNGs. In the literature, there are some guidelines and standards for health tests; however, there are deficiencies left to the user, or test definitions with unclear mathematical substructures are given. In this study, a mathematical model for health tests is introduced and constructed with some selected random variables. This study describes how to increase or decrease the number of random variables and significance levels and customize the test suite for a specific noise source.

Organization. Chapter 2 provides preliminaries on RNGs, entropy, and some statistical analysis techniques, chapter 3 presents detailed statistical analysis on NIST SP 800-90B containing evaluation for accuracy and correlations. Chapter 4 introduces the new entropy estimator *index-value coincidence estimate* in detail. Chapter 5 introduces the proposed health test suite with some mathematical backgrounds of random variables and experimental results. Chapter 6 presents conclusion of the thesis.

CHAPTER 2

PRELIMINARIES

2.1 Random Number Generators

Random numbers have a wide range of uses in scientific fields such as mathematics, statistics, modeling, biology, computer science, and cryptography. Random numbers are generated by random number generators (RNGs), a software algorithm or hardware mechanism that generates random number sequences. There are mainly two types of RNGs:

- True Random Number Generators (TRNGs)
- Pseudo Random Number Generators (PRNGs)

TRNG extracts randomness from physical phenomena, such as thermal noise, atmospheric noise, radioactive decay, free-running oscillator, or movement of an electron. A cryptographic TRNG is mainly composed of multiple components, including (i) a *noise source* (entropy source) that extracts randomness from physical phenomena to generate a *seed*, (ii) a *processing function* that extends the seed to generate a long random-looking sequence, and (iii) *health tests* that verify the ongoing functionality of the noise source. To increase the statistical quality and randomness of the data generated by noise source, processing functions are used; they are deterministic mathematical or cryptographic algorithms such as block ciphers, hash functions, etc. Since processing functions are deterministic, the entropy is solely provided by the noise source, and it is important to measure the unpredictability of the noise source outputs.

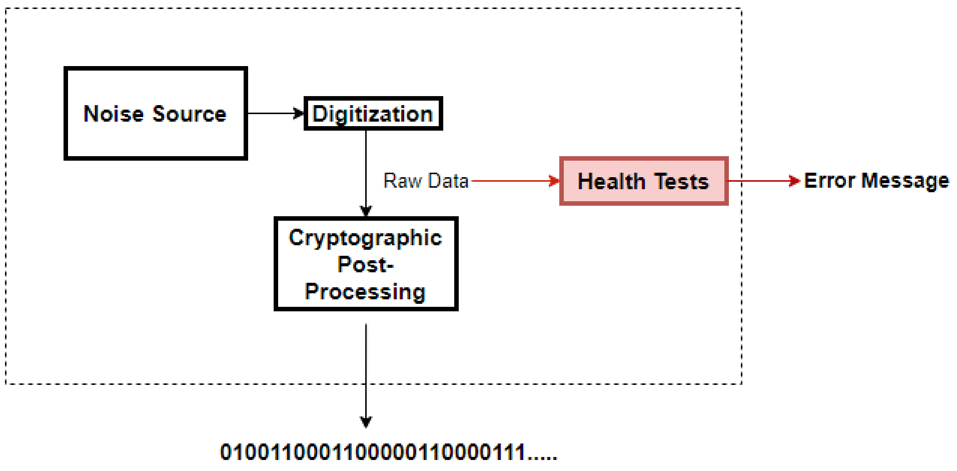

Figure 2.1: True Random Number Generator Model

PRNG (also known as a *deterministic random bit generator*) are deterministic algorithms that generate random-looking sequences from seeds. Linear-feedback shift registers or stream ciphers can be examples of PRNGs. Since the process is deterministic and by using the same seeds, the same sequences can be reproduced, the outputs of PRNGs are called pseudo-random number sequences. They are used for cryptographic purposes excessively because they reduce transmission and storage costs. Similarly, it is important to test the outputs to understand the unpredictability of the outputs.

Testing methodologies of RNGs can be categorized as follows:

1. **Statistical Randomness Tests:** The outputs of RNGs are tested by statistical randomness tests, which examine certain statistical characteristics of the output, compare the result to those in random sequences, and evaluate the outputs in terms of randomness. In general, test suites containing several number of statistical tests are constructed based on statistical methods such as chi-square goodness-of-fit test, etc.

2. **Entropy Estimation:** The amount of unpredictability or uncertainty extracted by the noise source of TRNG is evaluated by entropy estimation methods.

These tests focus on physical components of the mechanisms. There are statistical and prediction-based block-box entropy estimation methods, and also there are some stochastic models.

3. **Health Tests:** Health tests are integral parts of TRNGs, they are designed to detect corruption and error in the working mechanism of the entropy source and give warnings about unexpected changes, simultaneously. These tests are designed as algorithms, checking basic randomness properties of middle step outputs of TRNG, that provide fast results and have low time complexity.

In the content of this thesis, we focus on *entropy estimation* and *health tests*.

2.2 Entropy

In information theory, *entropy* was introduced by Claude Shannon, as a measure of uncertainty associated with the outcomes of a random variable [29].

To measure how much information is produced by observing a set of possible events whose probabilities of occurrence are $p_1, p_2, \cdots, p_n$, information function I can be characterized with the following properties:

- I is continuous in p_i's.
- If the probabilities of occurrences are equal; that is, for each i, $p_i = \frac{1}{n}$,then I is a monotonic increasing function of n.
- When a complex choice follows smaller, sequential choices, the overall information of the original choice can be calculated by adding up the information of each individual choice, weighted by their probabilities.

It was deduced that a function satisfying these properties may be of the form

$$I = -K \sum_{i=1}^{n} p_i \log p_i \tag{2.1}$$

where K is a positive constant. Then *entropy* was defined by Shannon [29] as follows:

Definition 2.2.1. *Let $\mathcal{X}$ be a random variable that takes values from the set $A = \{x_1, x_2, \ldots, x_n\}$ with probabilities $Pr(\mathcal{X} = x_i) = p_i$ for $i = 1, 2, \ldots, n$. The entropy of the random variable $\mathcal{X}$ is defined as*

$$H = -\sum_{i=1}^{n} p_i \log p_i \tag{2.2}$$

For example, when a fair coin is tossed, the probability of getting a Head and a Tail are equal and $\frac{1}{2}$. The entropy is:

$$H = -\sum_{i=1}^{2} p_i \log p_i \tag{2.3}$$

$$= -(\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2}) = 1 \tag{2.4}$$

If the coin is not fair and the probability of getting a Head is $\frac{1}{3}$ and a Tail is $\frac{2}{3}$. Then the entropy is evaluated as:

$$H = -\sum_{i=1}^{2} p_i \log p_i \tag{2.5}$$

$$= -(\frac{1}{3}\log\frac{1}{3} + \frac{2}{3}\log\frac{2}{3}) = 0.9 \tag{2.6}$$

In the second case, the tail is more likely to come up than the head, so the uncertainty is less than in the first case.

Rényi introduced a class of entropy functions [26],

Definition 2.2.2. *Let $\mathcal{X}$ be a random variable that takes values from the set $A = \{x_1, x_2, \ldots, x_n\}$ with probabilities $Pr(\mathcal{X} = x_i) = p_i$ for $i = 1, 2, \ldots, n$. The Rényi entropy of order-α of the random variable $\mathcal{X}$ is defined as*

$$H_\alpha = \frac{1}{1-\alpha} \log \big(\sum_{i=1}^{n} {p_i}^{\alpha}\big) \tag{2.7}$$

where $0 < \alpha < \infty$ and $\alpha \neq 1$.

Note that, the limit when $\alpha \to 1$, Rényi entropy gives the Shannon entropy and as $\alpha \to \infty$, the Rényi entropy is determined by the highest probability p_i in the distribution, this case is defined as *minimum entropy*.

In literature, there are several measures of entropy. In general, most conservative one *min-entropy* is used for the estimation of the entropy of RNGs.

Definition 2.2.3. *Let $\mathcal{X}$ be a random variable that takes values from the set $A = \{x_1, x_2, \dots, x_n\}$ with probabilities $Pr(\mathcal{X} = x_i) = p_i$ for $i = 1, 2, \dots, n$. The min-entropy of the random variable $\mathcal{X}$ is defined as*

$$\begin{aligned} H_\infty &= \min_{1 \le i \le n} (-\log_2 p_i) && (2.8) \\ &= -\log_2(\max_{1 \le i \le n} p_i). && (2.9) \end{aligned}$$

Entropy was defined as an average level of information or uncertainty associated with the outcomes of a random variable, this measure is used in many applications such as data compression, encoding, cryptanalysis, random number generation etc.

TRNGs typically rely on some physical processes (such as thermal noise, atmospheric noise, radioactive decay, or electronic noise) to generate random numbers. It is important to measure the quality and variability of the physical phenomena that TRNG relies on to contribute to the entropy. Entropy is used to evaluate noise sources of TRNGs, there are some statistical based black-box models and stochastic models to estimate the entropy of random number generators.

2.3 Correlation

Correlation is a measure that describes the statistical relation between two random variables. The Pearson [24] and Spearman [30] correlation coefficients are commonly used metrics to measure the correlation between two random variables. The correlation coefficients take values between -1 and 1. A value close to 1 or -1 shows a strong positive or negative association between variables, whereas a value close to 0 shows a weak association. See Table 2.1 for the interpretation of the Pearson r and Spearman correlation coefficients ρ.

Table 2.1: Interpretation of Pearson r and Spearman ρ correlation coefficients

Interval	**Interpretation**
$0 < \lvert r\rvert, \lvert\rho\rvert \leq 0.20$	Negligible correlation
$0.2 < \lvert r\rvert, \lvert\rho\rvert \leq 0.40$	Weak correlation
$0.4 < \lvert r\rvert, \lvert\rho\rvert \leq 0.60$	Moderate correlation
$0.6 < \lvert r\rvert, \lvert\rho\rvert \leq 0.80$	High correlation
$0.8 < \lvert r\rvert, \lvert\rho\rvert \leq 1$	Strong correlation

The Pearson correlation [24] measures the strength of a linear relationship between two random variables, assuming that the variables are distributed normally, whereas the Spearman correlation [30] describes the monotonic relationship between variables without the assumption that the variables have normal distribution.

Definition 2.3.1. *Let $\mathcal{X}$ and $\mathcal{Y}$ be random variables. The Pearson correlation coefficient r between a given paired dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is defined as*

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{2.10}$$

where n is the sample size, x_i and y_i are sample points, $\bar{x}$ is the sample mean of $\mathcal{X}$, and $\bar{y}$ is the sample mean of $\mathcal{Y}$.

Definition 2.3.2. *Let $\mathcal{X}$ and $\mathcal{Y}$ be random variables. The Spearman correlation coefficient ρ between a given paired dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ is defined as*

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}, \tag{2.11}$$

where n is the sample size, and d_i is the difference between the rank of the paired samples.

In this thesis, mutual correlations between entropy estimators on NIST SP 800-90B are calculated according to Pearson's and Spearman's metrics. For necessary cases, to control the false discovery rate, the Benjamini-Hochberg procedure [5] was applied to interpret the results. We had multiple hypotheses regarding the correlations between the tests. Therefore, we adjusted the P-values using Benjamini-Hochberg procedure in order to reduce the false positive outcomes.

CHAPTER 3

OBSERVATIONS ON NIST SP 800-90B

In this chapter, the effectiveness and limitations of the NIST SP 800-90 methods are evaluated by exploring the accuracy of these estimators using simulated random numbers with known entropy, investigating the correlation between entropy estimates, studying the impacts of deterministic transformations on the estimators and impact of IID-assumption tests on estimators.

3.0.1 Entropy Estimation Based on NIST SP 800-90B

NIST SP 800-90B [32] describes an *entropy source* model, that is composed of a noise source, health tests, and an optional conditioning function. The standard also provides guidelines for the generation of random numbers using entropy sources and specifies entropy estimation techniques to ensure the randomness and unpredictability of the outputs. These black-box techniques are applied to noise source outputs and are independent of the internals of the noise source.

NIST SP 800-90B [32] defines two tracks to estimate the min-entropy of an entropy source: independent and identically distributed (IID) and non-IID. To determine which track to use, a number of statistical tests are applied to an output sequence generated by the entropy source to check the IID assumption. If the output sequence passes these tests, the source is assumed to generate IID outputs, and only the most common value method is used to estimate the entropy. Otherwise, the source is assumed to generate non-IID outputs, and the minimum of the 10 NIST SP 800-90B estimators is used to estimate the entropy of the source.

NIST SP 800-90B [32] is testing IID assumption with some statistical tests based on permutation (shuffling) tests and chi-square tests. For permutation tests, Fisher-Yates shuffle algorithm is used to generate $10\,000$ permutations of the given sequence. NIST SP 800-90B defines 11 test statistics for permutation tests. According to a test statistic, the results of the original sequence and the results of permutations are compared. Test statistics are listed with their evaluation metrics in Table 3.1.

Excursion, Number of Runs Based on Median, Length of Runs Based on Median, and Compression test statistics are directly employed for binary sequences. For applying the remaining test statistics, binary sequences are turned into integer sequences by defined conversions.

Table 3.1: Test statistics of Permutation Tests of NIST SP 800-90B

Test Statistic	***Metric***	Directly applied to binary
Excursion	How far the running sum of samples deviates from average	✓
Number of Directional Runs	The number of runs constructed using the relations between consecutive samples	×
Length of Directional Runs	The length of longest run constructed using the relations between consecutive samples	×
Number of Increases and Decreases	The maximum number of increases or decreases between consecutive samples	×
Number of Runs Based on Median	The number of runs that are constructed with respect to the median of the sequence	✓
Length of Runs Based on Median	The length of the longest run that is constructed with respect to the median of the sequence	✓
Average Collision	The average of the number of successive sample values until a duplicate is found	×
Maximum Collision	The maximum of the number of successive sample values until a duplicate is found	×
Periodicity	The number of periodic structures in the sequence	×
Covariance	Measures the strength of the lagged correlation	×
Compression	Measures length of the compressed string	✓

In NIST SP 800-90B [32], additional Chi-square statistical tests, defined for testing independence and goodness-of-fit are also in IID-Assumption tests. The length of the longest repeated substring test is also employed in the IID-Assumption part.

NIST SP 800-90B [32] describes ten entropy estimators: most common value, collision, Markov, compression, t-tuple, longest repeated substring (LRS), multi most common in window prediction, lag prediction, multiple Markov Model with Counting (multiMMC) prediction, and LZ78Y. The minimum of these ten estimates is used to estimate the min-entropy of the noise source outputs. Table 3.2 lists the estimators and corresponding metrics provided in the standard. Some of the estimators, namely collision, Markov, and compression, are only defined for binary inputs (i.e., $n = 2$).

The estimators take noise source outputs $S = (s_1, s_2, \ldots, s_L)$, where $s_i \in A = \{x_1, x_2, \ldots, x_n\}$ and return an min-entropy estimate between 0 and $\log_2 n$. Note that to establish the final entropy estimate, the standard additionally considers the entropy estimate from the designers, and the impact of the conditioning components, etc.

Table 3.2: Entropy Estimators of NIST SP 800-90B

Estimator	***Metric***	Support for $n > 2$?
Most Common Value	Proportion of the most common value in the input data set	✓
Collision	Probability of the most-likely output, depending on the number of collisions	$\times$
Markov	Dependencies between consecutive values	$\times$
Compression	Compression amount of the input dataset	$\times$
t-Tuple	Frequency of t-tuples	✓
Longest Repeated Substring (LRS)	Number of repeated substrings	✓
Multi Most Common in Window Prediction	Number of correct predictions based on the most common value	✓
Lag Prediction	Number of correct predictions based on periodicity	✓
MultiMMC Prediction	Number of correct predictions based on multiple Markov models	✓
LZ78Y Prediction	Number of correct predictions based on a dictionary constructed using observed tuples	✓

3.1 Methodology

The goal of this study is to answer the following questions regarding the entropy estimators introduced in SP 800-90B [32]:

1. *How closely do the entropy estimators match the true entropy of the source?*
2. *How correlated are the entropy estimators?*
3. *How do different deterministic transformations impact the entropy estimate?*
4. *How do IID-Assumption test results impact the entropy estimate?*

3.1.1 Entropy Estimation using Known Distributions

One approach to understanding the accuracy of the entropy estimators is to simulate various sequences with known probability distributions (hence, known entropy), and check the difference between the estimated entropy and the true entropy. In cases where certain entropy estimators consistently yield outlier results compared to others, it is important to investigate the underlying reasons for such discrepancies. This could involve examining the specific characteristics of the input data, inherent biases in the estimation techniques, or the impacts of using different input lengths and sample sizes.

3.1.2 Correlation of the Entropy Estimators

Understanding the correlation between different entropy estimators can provide insights into the reliability, robustness, and limitations of the estimators for cryptographic applications. One aspect to consider is the agreement between different entropy estimation methods by assessing whether they tend to produce similar entropy estimates for the same set of input sequences. This study employed correlation analysis to quantify the relationship between pairs of entropy estimates and used the Pearson and Spearman correlation coefficients.

3.1.3 Impact of Deterministic Transformations

The noise source outputs are typically processed using deterministic conditioning functions to reduce their statistical bias and improve their entropy rate (i.e., entropy per bit). The impacts of a number of deterministic transformations that are applied to the output sequence are of interest here.

Let $S = (s_1, s_2, \ldots, s_L)$ be a noise source output with length L, and let $S' = (s'_1, s'_2, \ldots, s'_L)$ be generated from S via a deterministic transformation. This study uses the following transformations:

- **Reverse:** This transformation generates a new sequence by changing the order of the sequence. The generated sequence $S' = (s_L, s_{L-1}, \ldots, s_2, s_1)$ is constructed with $s'_i = s_{L-i+1}$ for each $i = 1, 2, \ldots, L$.
  For example, the reversed sequence of $S = (\texttt{10110001110010})$ is
  $S' = (\texttt{01001110001101})$.

- **Binary Derivative:** This transformation generates a new sequence by XORing (i.e., modulo 2 addition) the consecutive bits of the sequence. The generated sequence $S' = (s'_1, s'_2, \ldots, s'_L)$ is constructed with

$$s'_i = \begin{cases} s_i \oplus s_{i+1}, & i = 1, 2, \ldots, L-1, \\ s_1, & i = L. \end{cases}$$

  For example, the binary derivative of $S = (\texttt{10110001110010})$ is
  $S' = (\texttt{11010010010111})$.

- **t-Rotation:** This transformation applies a t-bit rotation to the input sequence, i.e., t-bit rotation of the sequence S where $S = (s_1, s_2, \ldots, s_L)$ is
  $S' = (s_{t+1}, s_{t+2}, \ldots, s_L, s_1, s_2, \ldots, s_t)$, where $t = 16, 64, 128$, or 1024.
  For example, 2-bit rotation of $S = (\texttt{10110001110010})$ is
  $S' = (\texttt{11000111001010})$.

3.1.4 Entropy Estimation According to IID-Assumption Tests using Known Distributions

To understand the effect of IID-Assumption tests on the entropy estimation, we simulate various sequences with known probability distributions (hence, known entropy), and check the difference between the estimated entropy and the true entropy, when estimations are done according to IID-Assumption tests results.

3.2 Experimental Results

3.2.1 Accuracy of Entropy Estimators

Simulated Datasets

The following datasets with known entropy were simulated for the experiments to measure accuracy of entropy estimators:

1. **Uniform distribution with full entropy.** The datasets are generated using the Cipher Block Chaining (CBC) mode of the block cipher Advanced Encryption Standard (AES) [9]. Sequences are generated for three different sample sizes (i.e., the size of the noise source output): binary, 4-bit, and 8-bit. For each sample size, 1000 sequences of length $1\,000\,000$ were generated. In these sequences, all outputs are assumed to have an equal probability of occurring, and are independent. Hence, the outputs have full entropy.

2. **Biased binary distribution with entropy=0.5.** The dataset follows a biased binary distribution, where the probability of observing a 1 is 0.7, and the probability of observing a 0 is 0.3. For each sample size, 1000 sequences of length $1\,000\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator Mersenne Twister (MT19937) in `C++`.

3. **4-bit near-uniform with entropy=0.5.** This dataset follows a 4-bit near-uniform distribution, where the probability of observing the template `0000` is 0.25, and the probability of observing other 4-bit templates is 0.05. For each sample size, 1000 sequences of length $250\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

4. **8-bit near-uniform with entropy=0.5.** This dataset follows an 8-bit near-uniform distribution, where the probability of observing the template `00000000` is 0.06, and the probability of observing other 8-bit templates is 0.003686. For each sample size, 1000 sequences of length $125\,000$ were generated. In these

sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

Table 3.3 compares the actual and estimated entropy values for binary, 4-bit, and 8-bit uniformly distributed data with full entropy. It shows that compression and collision estimates produce the smallest estimates for binary data, which is consistent with the findings of Zhu et al. [35] and Kim et al. [17].

Table 3.3: Mean and standard deviation of entropy estimators for binary, 4-bit, and 8-bit sources with full entropy

	1-bit		**4-bit**			**8-bit**		
	Mean	**Std. Dev.**	**Mean**	**Mean/bit**	**Std. Dev.**	**Mean**	**Mean/bit**	**Std. Dev.**
MCV	0.9951	0.0009	3.9514	0.9879	0.0056	7.6736	0.9592	0.0222
Collision	0.9141	0.0194	*	*	*	*	*	*
Markov	0.9982	0.0011	*	*	*	*	*	*
Compression	0.8535	0.0287	*	*	*	*	*	*
t-Tuple	0.9294	0.0104	3.7799	0.9450	0.0149	7.6736	0.9592	0.0222
LRS	0.9785	0.0262	3.8928	0.9732	0.1131	7.7468	0.9683	0.1878
Multi MCW	0.9954	0.0114	3.9635	0.9909	0.0662	7.8169	0.9771	0.1315
Lag Prediction	0.9957	0.0072	3.9677	0.9919	0.0416	7.8116	0.9764	0.1679
MultiMMC	0.9951	0.0129	3.9616	0.9904	0.0778	7.8197	0.9775	0.1302
LZ78Y	0.9956	0.0096	3.9616	0.9904	0.0778	7.8198	0.9775	0.1302

The same experiments were repeated for biased binary distribution, 4-bit near-uniform distribution, and 8-bit near-uniform distribution, and the results are summarized in Table 3.4. Similar to uniform distribution, the compression estimate underestimates entropy for biased distributions. However, LRS and lag prediction overestimate the entropy by approximately 50%. Similar results were obtained for 4-bit and 8-bit samples.

Table 3.4: Mean and standard deviation of entropy estimators of datasets for biased binary, 4-bit near-uniform, and 8-bit near-uniform distributions

	Biased Binary Dist.		**4-bit Near-uniform**			**8-bit Near-uniform**		
	Mean	**Std. Dev.**	**Mean**	**Mean/bit**	**Std. Dev.**	**Mean**	**Mean/bit**	**Std. Dev.**
MCV	0.5122	0.0009	1.9872	0.4968	0.0050	4.0169	0.5021	0.0160
Collision	0.5095	0.0020	*	*	*	*	*	*
Markov	0.5146	0.0011	*	*	*	*	*	*
Compression	0.3224	0.0009	*	*	*	*	*	*
t-Tuple	0.5031	0.0116	1.9710	0.4928	0.0197	3.9993	0.4999	0.0380
LRS	0.7692	0.0205	3.2364	0.8091	0.0954	6.9466	0.8683	0.1884
Multi MCW	0.5121	0.0055	1.9860	0.4965	0.0200	4.0063	0.5008	0.0738
Lag Prediction	0.7756	0.0263	3.2812	0.8203	0.0923	6.9558	0.8695	0.2984
MultiMMC	0.5118	0.0055	1.9861	0.4965	0.0200	4.1557	0.5195	0.1028
LZ78Y	0.5118	0.0055	1.9860	0.4965	0.0200	4.1556	0.5194	0.1027

Figures 3.1, 3.2 and 3.3 show the distribution of the entropy estimation for uniform distribution with full entropy in binary, 4-bit and 8-bit representations. Compression and LRS estimators seem to show high variation compared to other estimators.

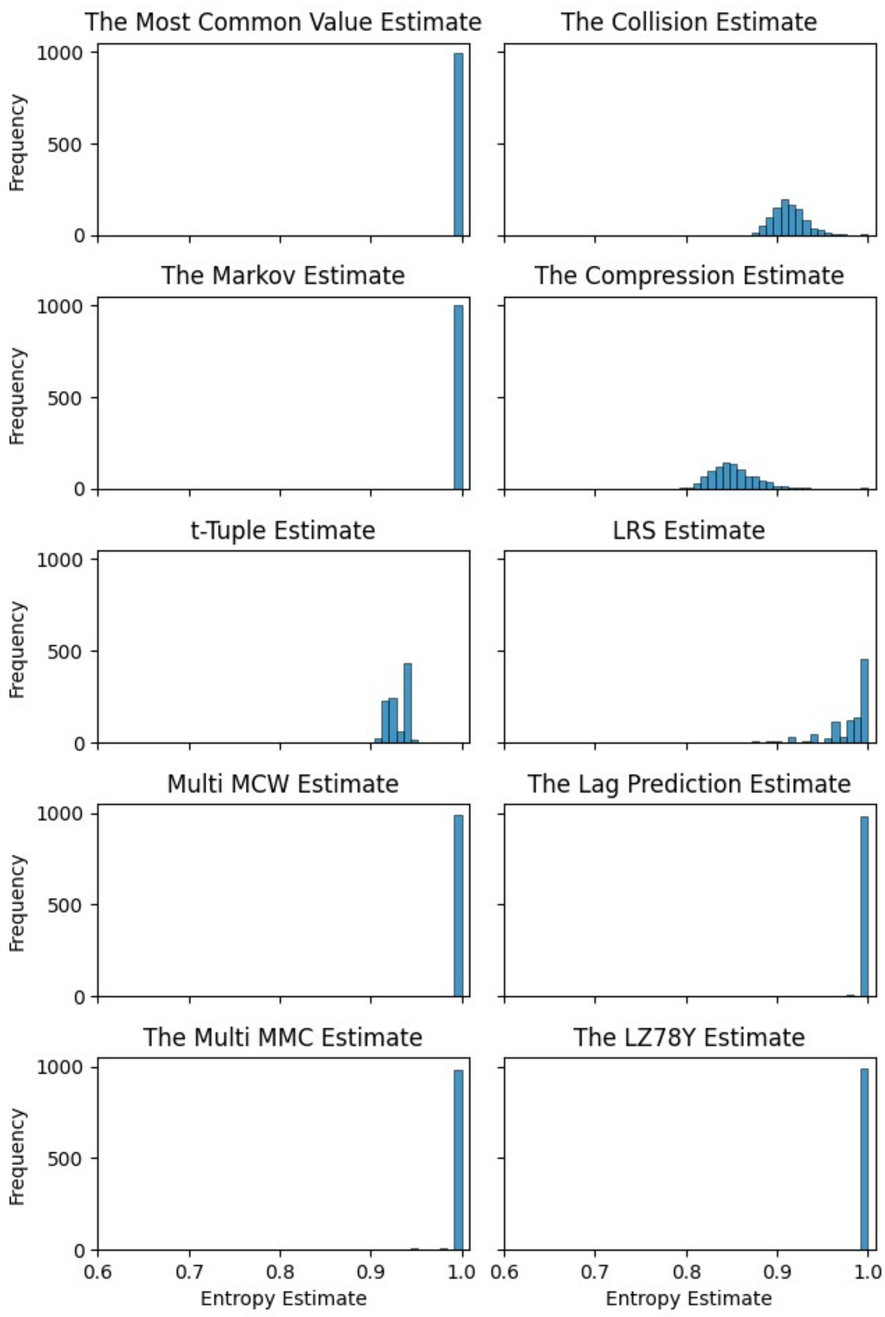

Figure 3.1: Distribution of Entropy Estimations for Binary Uniform Distribution Dataset

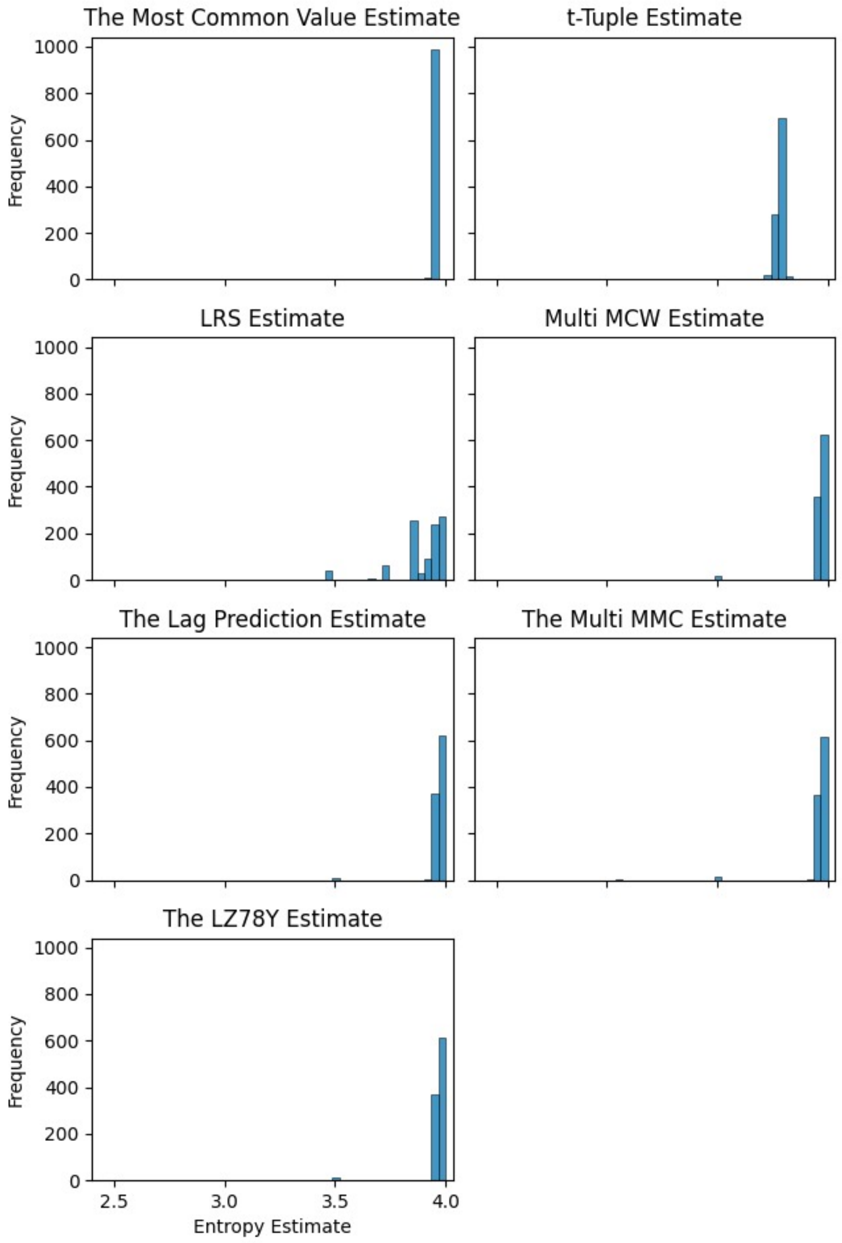

Figure 3.2: Distribution of Entropy Estimations for 4-bit Uniform Distribution Dataset

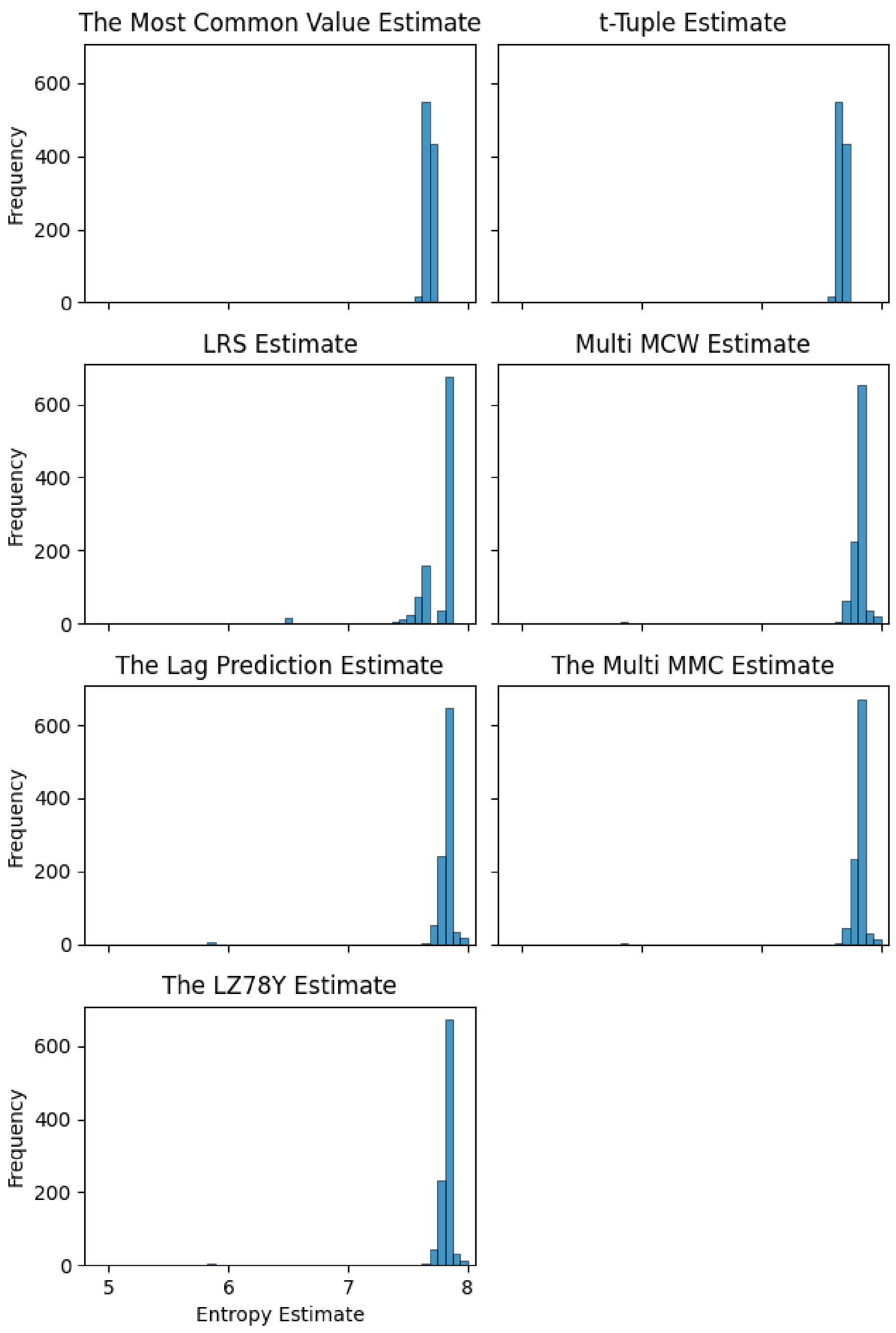

Figure 3.3: Distribution of Entropy Estimations for 8-bit Uniform Distribution Dataset

3.2.2 Correlations of Estimators

The Pearson and Spearman coefficients were used to measure the correlation between entropy estimators. To analyze correlation of the estimators mainly three different datasets are used in experiments: IID-Full Entropy, IID-Low Entropy and Non-IID-Low Entropy.

3.2.2.1 Correlation Analysis with Dataset 1: IID-Full Entropy

For these experiments, sequences that have uniform distribution with full entropy are generated in different lengths. To generate these sequences Cipher Block Chaining (CBC) mode of the block cipher Advanced Encryption Standard (AES) [9] is used. For each length, $1\,000\,000$, $8\,000\,000$, and $32\,000\,000$ bits, 200 binary sequences are generated to construct datasets. In these sequences, all outputs are assumed to have an equal probability of occurring, and are independent and identical. Hence, the outputs have full entropy.

Using 200 binary sequences of length $1\,000\,000$, Table 3.5 and Table 3.6 show the Pearson and Spearman correlations among different estimators, respectively. According to Table 3.5, a strong or moderate correlation was observed for the (MCV, Markov), (MultiMCW, MultiMMC) (MultiMMC, LZ78Y), and (MultiMCW, LZ78Y) estimators using Pearson's metric. When the same experiments were conducted using Spearman's metric, a correlation was still observed between (MCV, Markov). However, (MultiMMC, LZ78Y) and (MultiMCW, LZ78Y) correlations were no longer as strong. Additionally, the correlation between (Markov, LZ78Y) was observed to be strong for Spearman's metric.

Table 3.5: Pearson correlation among different estimators for uniform distribution with full entropy $1\,000\,000$

	MCV	**Collision**	**Markov**	**Compression**	**t-Tuple**	**LRS**	**MultiMCW**	**Lag Prediction**	**MultiMMC**	**LZ78Y**
MCV	1.0000	-0.0531	**0.5338**	-0.1170	0.0564	-0.0506	0.0535	-0.0745	0.2174	0.2610
Collision		1.0000	0.1315	-0.0092	0.0163	0.0563	0.0071	-0.0281	-0.0286	-0.0856
Markov			1.0000	0.0347	0.0821	-0.0158	0.0261	-0.0581	0.1767	0.2278
Compression				1.0000	-0.0422	0.0284	0.0281	-0.0011	0.1094	0.0756
t-Tuple					1.0000	0.0388	0.0444	0.0583	0.0760	0.0765
LRS						1.0000	-0.0449	0.0059	-0.0557	-0.0505
MultiMCW							1.0000	-0.0063	**0.4702**	**0.8063**
Lag Prediction								1.0000	-0.0363	-0.0281
MultiMMC									1.0000	**0.4693**
LZ78Y										1.0000

Table 3.6: Spearman correlation among different estimators for uniform distribution with full entropy 1 000 000

	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	-0.0426	**0.5410**	-0.1012	0.0636	-0.0317	-0.0601	0.0314	0.1825	0.4991
Collision		1.0000	0.1224	0.0282	0.0254	0.0035	0.0140	0.0009	0.0017	-0.1207
Markov			1.0000	0.0491	0.0954	-0.0215	-0.0454	0.0510	0.1784	**0.6420**
Compression				1.0000	0.0138	0.1014	0.0202	0.0200	0.1711	0.1143
t-Tuple					1.0000	0.0714	-0.0104	-0.0789	0.0316	0.0575
LRS						1.0000	0.0396	-0.0641	0.0187	0.0008
MultiMCW							1.0000	-0.0593	0.0784	-0.1028
Lag Prediction								1.0000	0.0178	0.1391
MultiMMC									1.0000	0.1982
LZ78Y										1.0000

To observe the effect of lengths of sequences on estimations, the same experiments were repeated for the dataset containing binary sequences of lengths 8 000 000 and 32 000 000.

Table 3.7: Pearson correlation among different estimators for uniform distribution with full entropy 8 000 000

	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	-0.0160	0.3343	-0.0555	0.1460	0.1293	0.0098	0.0315	0.0214	0.0680
Collision		1.0000	0.2238	-0.0835	0.0917	-0.0557	0.0650	0.0007	0.0433	0.0832
Markov			1.0000	-0.0476	0.0726	-0.0071	-0.1051	0.0386	-0.0178	0.0081
Compression				1.0000	0.0197	-0.0021	-0.0127	-0.0228	0.0042	0.0017
t-Tuple					1.0000	-0.0032	0.0425	0.0275	0.0664	0.1016
LRS						1.0000	-0.0047	0.0789	0.0231	0.0192
MultiMCW							1.0000	-0.0109	0.5500	0.7687
Lag Prediction								1.0000	-0.0108	-0.0122
MultiMMC									1.0000	0.7118
LZ78Y										1.0000

Table 3.8: Spearman correlation among different estimators for uniform distribution with full entropy 8 000 000

	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	-0.0092	0.3450	-0.0659	0.0857	0.0821	0.0196	-0.0024	0.0706	0.4064
Collision		1.0000	0.2706	-0.0932	0.0568	-0.0140	0.1265	-0.0027	-0.0157	0.0125
Markov			1.0000	-0.0429	0.0238	0.0454	0.0389	-0.0027	0.1123	0.6106
Compression				1.0000	0.0114	0.0006	-0.0461	-0.0108	-0.0405	-0.0579
t-Tuple					1.0000	0.0321	-0.0927	-0.0706	0.0345	0.0993
LRS						1.0000	0.0638	0.1122	0.0429	0.0073
MultiMCW							1.0000	0.1930	0.0388	-0.0076
Lag Prediction								1.0000	-0.0254	-0.0492
MultiMMC									1.0000	0.1859
LZ78Y										1.0000

Table 3.9: Pearson correlation among different estimators for uniform distribution with full entropy 32 000 000

	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	-0.1224	0.4393	-0.0279	-0.0736	0.0506	0.0639	-0.0673	0.0568	0.0209
Collision		1.0000	0.0124	0.0093	-0.0065	-0.0077	-0.2205	0.0820	0.0428	0.0258
Markov			1.0000	-0.0902	-0.0818	0.0311	0.0106	-0.0152	-0.0300	0.0001
Compression				1.0000	-0.0554	-0.0108	0.0170	-0.1724	-0.0933	-0.0132
t-Tuple					1.0000	0.0999	0.0355	0.0082	-0.0411	-0.1034
LRS						1.0000	0.0817	-0.0683	-0.0427	-0.0707
MultiMCW							1.0000	-0.0148	0.1119	0.1361
Lag Prediction								1.0000	-0.0087	-0.0097
MultiMMC									1.0000	0.3900
LZ78Y										1.0000

Table 3.10: Spearman correlation among different estimators for uniform distribution with full entropy 32 000 000

	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	-0.0898	0.3699	-0.0439	-0.0499	0.0501	0.0495	0.0115	0.1404	0.4301
Collision		1.0000	0.0980	-0.0553	0.0094	0.0765	0.1190	-0.0474	-0.1012	-0.1570
Markov			1.0000	-0.1085	-0.0427	0.0173	0.1167	0.0002	0.2001	0.6187
Compression				1.0000	0.0025	-0.0019	0.0593	0.0052	0.0558	-0.0178
t-Tuple					1.0000	0.0406	-0.0640	0.0304	0.0054	-0.0866
LRS						1.0000	0.0632	-0.0315	-0.0204	0.0290
MultiMCW							1.0000	0.0546	0.0963	0.1163
Lag Prediction								1.0000	0.0984	0.0236
MultiMMC									1.0000	0.3277
LZ78Y										1.0000

Experiments show that there is a moderate correlation between (MCV, Markov) according to Pearson's and also Spearman's metric. When the length of sequences increases to 8 million and 32 million, we observe a small decrease in correlation coefficients of (MCV, Markov) compared to 1 million cases for each metric.

Similar to initial results, for sequences of length 8 million, a strong or moderate correlation was observed for (MultiMCW, MultiMMC) (MultiMMC, LZ78Y) and (MultiMMC, LZ78Y) according to Pearson. But we repeat the same experiments with sequences of length 32 million, Pearson correlations can be interrupted as low. According to Spearman's metric, it was not observed considerable correlations between these estimators.

For each case (1m, 8m, 32m) correlation between (Markov, LZ78Y) was observed to be strong for Spearman's metric.

Even if a small decrease in the correlation coefficients was observed when the experiments were repeated, the general results of experiments are close to each other.

3.2.2.2 Correlation Analysis with Dataset 2: IID-Low Entropy

Experiments were repeated with biased datasets, to observe the relations of the estimators when sequences have not full entropy. The following datasets were used for the experiments, similarly Pearson and Spearman coefficients were used to measure the correlation between entropy estimators. However, the number of highly correlated estimators is seen as the result of experiments. To make accurate observations Benjamini-Hochberg correction [5] applied to the P-values, $p < 0.01$ is assumed to be significant.

1. **Biased binary distribution with entropy=0.5.** The dataset follows a biased binary distribution, where the probability of observing a 1 is 0.7, and the probability of observing a 0 is 0.3. 200 sequences of length $1\,000\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator Mersenne Twister (MT19937) in `C++`.

2. **4-bit near-uniform with entropy=0.5.** This dataset follows a 4-bit near-uniform distribution, where the probability of observing the template `0000` is 0.25, and the probability of observing other 4-bit templates is 0.05. 200 sequences of length $1\,000\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

3. **8-bit near-uniform with entropy=0.5.** This dataset follows an 8-bit near-uniform distribution, where the probability of observing the template `00000000` is 0.06, and the probability of observing other 8-bit templates is 0.003686. 200 sequences of length $1\,000\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

The Pearson and Spearman correlations of the estimators are given in the following tables.

When we interpret Pearson correlation results of estimators for biased binary sequences, it was observed a strong correlation for (Markov, MCV), (Compression, MCV) and (Markov, Collision). There was a moderate correlation between the pairs (Collision, MCV) and (Compression, Markov).

Table 3.11: Pearson correlation among different estimators for binary biased distribution with 0.5 entropy

Pearson	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.4903	0.8381	0.7580	0.1988	0.0359	0.1706	0.2058	0.1718	0.1718
Collision	**0.4903**	1.0000	0.7173	0.2839	0.1616	0.0434	0.0599	0.1563	0.0612	0.0611
Markov	**0.8381**	**0.7173**	1.0000	0.5893	0.2238	0.0295	0.1551	0.1846	0.1561	0.1561
Compression	**0.7580**	**0.2839**	**0.5893**	1.0000	0.1558	0.0493	0.1396	0.1281	0.1403	0.1404
t-Tuple	0.1988	0.1616	**0.2238**	0.1558	1.0000	0.0882	0.1637	0.1478	0.1637	0.1636
LRS	0.0359	0.0434	0.0295	0.0493	0.0882	1.0000	0.3244	-0.0036	0.3241	0.3240
MultiMCW	0.1706	0.0599	0.1551	0.1396	0.1637	**0.3244**	1.0000	0.0114	1.0000	1.0000
Lag Prediction	**0.2058**	0.1563	0.1846	0.1281	0.1478	-0.0036	0.0114	1.0000	0.0115	0.0115
MultiMMC	0.1718	0.0612	0.1561	0.1403	0.1637	**0.3241**	**1.0000**	0.0115	1.0000	1.0000
LZ78Y	0.1718	0.0611	0.1561	0.1404	0.1636	**0.3240**	**1.0000**	0.0115	**1.0000**	1.0000

Table 3.12: P-values of Pearson correlation among different estimators for binary biased distribution with 0.5 entropy

P-values	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.0000	0.0000	0.0000	0.0120	0.6819	0.0327	0.0091	0.0326	0.0326
Collision	**0.0000**	0.0000	0.0000	0.0001	0.0397	0.6151	0.4754	0.0429	0.4754	0.4754
Markov	**0.0000**	**0.0000**	0.0000	0.0000	0.0040	0.7375	0.0429	0.0212	0.0429	0.0429
Compression	**0.0000**	**0.0001**	**0.0000**	0.0000	0.0429	0.5672	0.0658	0.0930	0.0658	0.0658
t-Tuple	0.0120	0.0397	**0.0040**	0.0429	0.0000	0.2745	0.0382	0.0540	0.0382	0.0382
LRS	0.6819	0.6151	0.7375	0.5672	0.2745	0.0000	0.0000	0.9599	0.0000	0.0000
MultiMCW	0.0327	0.4754	0.0429	0.0658	0.0382	**0.0000**	0.0000	0.8906	0.0000	0.0000
Lag Prediction	**0.0091**	0.0429	0.0212	0.0930	0.0540	0.9599	0.8906	0.0000	0.8906	0.8906
MultiMMC	0.0326	0.4754	0.0429	0.0658	0.0382	**0.0000**	**0.0000**	0.8906	0.0000	0.0000
LZ78Y	0.0326	0.4754	0.0429	0.0658	0.0382	**0.0000**	**0.0000**	0.8906	**0.0000**	0.0000

According to Spearman's metric, there was a strong correlation between MCV and the estimators Markov, Compression, MultiMCW, MultiMMC, and LZ78Y. Similarly, it was observed that Compression is highly correlated with MultiMCW, MultiMMC, and LZ78Y. As a result, mutual correlations of MultiMCW, MultiMMC, and LZ78Y are very strong.

Table 3.13: Spearman correlation among different estimators for binary biased distribution with 0.5 entropy

Spearman	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.4518	0.8213	0.7377	0.2808	0.1646	0.9479	0.5232	0.9482	0.9481
Collision	**0.4518**	1.0000	0.6964	0.2680	0.1991	0.0926	0.4152	0.2021	0.4154	0.4162
Markov	**0.8213**	**0.6964**	1.0000	0.5732	0.3325	0.1354	0.7795	0.4046	0.7796	0.7804
Compression	**0.7377**	**0.2680**	**0.5732**	1.0000	0.2381	0.1468	0.7174	0.3923	0.7177	0.7180
t-Tuple	**0.2808**	**0.1991**	**0.3325**	**0.2381**	1.0000	0.1299	0.3130	0.1604	0.3126	0.3134
LRS	0.1646	0.0926	0.1354	0.1468	0.1299	1.0000	0.1964	0.0569	0.1959	0.1954
MultiMCW	**0.9479**	**0.4152**	**0.7795**	**0.7174**	**0.3130**	**0.1964**	1.0000	0.4887	0.9997	0.9996
Lag Prediction	**0.5232**	0.2021	**0.4046**	**0.3923**	0.1604	0.0569	**0.4887**	1.0000	0.4889	0.4894
MultiMMC	**0.9482**	**0.4154**	**0.7796**	**0.7177**	**0.3126**	**0.1959**	**0.9997**	**0.4889**	1.0000	0.9999
LZ78Y	**0.9481**	**0.4162**	**0.7804**	**0.7180**	**0.3134**	**0.1954**	**0.9996**	**0.4894**	**0.9999**	1.0000

Table 3.14: P-values of Spearman correlation among different estimators for binary biased distribution with 0.5 entropy

P-values	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.0000	0.0000	0.0000	0.0001	0.0225	0.0000	0.0000	0.0000	0.0000
Collision	**0.0000**	0.0000	0.0000	0.0002	0.0059	0.1963	0.0000	0.0053	0.0000	0.0000
Markov	**0.0000**	**0.0000**	0.0000	0.0000	0.0000	0.0595	0.0000	0.0000	0.0000	0.0000
Compression	**0.0000**	**0.0002**	**0.0000**	0.0000	0.0009	0.0413	0.0000	0.0000	0.0000	0.0000
t-Tuple	**0.0001**	**0.0059**	**0.0000**	**0.0009**	0.0000	0.0696	0.0000	0.0259	0.0000	0.0000
LRS	0.0225	0.1963	0.0595	0.0413	0.0696	0.0000	0.0065	0.4239	0.0065	0.0065
MultiMCW	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0065**	0.0000	0.0000	7.1427e-320	0.0000
Lag Prediction	**0.0000**	0.0053	**0.0000**	**0.0000**	0.0259	0.4239	**0.0000**	0.0000	0.0000	0.0000
MultiMMC Estimate	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0065**	**7.1427e-320**	**0.0000**	0.0000	0.0000
LZ78Y	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0065**	**0.0000**	**0.0000**	**0.0000**	0.0000

When we interrupt Pearson correlation results of estimators for biased 4-bit sequences, mutual correlations of MultiMCW, MultiMMC, and LZ78Y are very strong. Spearman's metric verified the same results. In addition to them, MCV was highly correlated with the same estimators MultiMCW, MultiMMC, and LZ78Y.

Table 3.15: Pearson correlation among different estimators for 4-bit biased distribution with 0.5 entropy

Pearson	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.0812	-0.0688	0.1238	0.0485	0.1239	0.1237
t-Tuple	0.0812	1.0000	0.0094	0.0749	0.1581	0.0750	0.0748
LRS	-0.0688	0.0094	1.0000	0.0051	0.0055	0.0054	0.0054
MultiMCW	0.1238	0.0749	0.0051	1.0000	-0.0147	1.0000	1.0000
Lag Prediction	0.0485	0.1581	0.0055	-0.0147	1.0000	-0.0145	-0.0146
MultiMMC	0.1239	0.0750	0.0054	**1.0000**	-0.0145	1.0000	1.0000
LZ78Y	0.1237	0.0748	0.0054	**1.0000**	-0.0146	**1.0000**	1.0000

Table 3.16: P-values of Pearson correlation among different estimators for 4-bit biased distribution with 0.5 entropy

P-values	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.4946	0.5265	0.1891	0.7356	0.1891	0.1891
t-Tuple	0.4946	0.0000	0.9431	0.4946	0.0828	0.4946	0.4946
LRS	0.5265	0.9431	0.0000	0.9431	0.9431	0.9431	0.9431
MultiMCW	0.1891	0.4946	0.9431	0.0000	0.9431	0.0000	0.0000
Lag Prediction	0.7356	0.0828	0.9431	0.9431	0.0000	0.9431	0.9431
MultiMMC	0.1891	0.4946	0.9431	**0.0000**	0.9431	0.0000	0.0000
LZ78Y	0.1891	0.4946	0.9431	**0.0000**	0.9431	**0.0000**	0.0000

Table 3.17: Spearman correlation among different estimators for 4-bit biased distribution with 0.5 entropy

Spearman	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.1697	0.0458	0.9653	0.3198	0.9654	0.9653
t-Tuple	0.1697	1.0000	0.0903	0.1860	0.1352	0.1855	0.1848
LRS	0.0458	0.0903	1.0000	0.0469	-0.0114	0.0460	0.0456
MultiMCW	**0.9653**	0.1860	0.0469	1.0000	0.2888	0.9996	0.9996
Lag Prediction	**0.3198**	0.1352	-0.0114	**0.2888**	1.0000	0.2924	0.2913
MultiMMC	**0.9654**	0.1855	0.0460	**0.9996**	**0.2924**	1.0000	0.9999
LZ78Y	**0.9653**	0.1848	0.0456	**0.9996**	**0.2913**	**0.9999**	1.0000

Table 3.18: P-values of Spearman correlation among different estimators for 4-bit biased distribution with 0.5 entropy

P-values	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.0228	0.5439	0.0000	0.0000	0.0000	0.0000
t-Tuple	0.0228	0.0000	0.2559	0.0131	0.0747	0.0131	0.0131
LRS	0.5439	0.2559	0.0000	0.5439	0.8732	0.5439	0.5439
MultiMCW	**0.0000**	0.0131	0.5439	0.0000	0.0001	0.0000	0.0000
Lag Prediction	**0.0000**	0.0747	0.8732	**0.0001**	0.0000	0.0001	0.0001
MultiMMC	**0.0000**	0.0131	0.5439	**0.0000**	**0.0001**	0.0000	0.0000
LZ78Y	**0.0000**	0.0131	0.5439	**0.0000**	**0.0001**	0.0000	0.0000

When we repeated the same experiments for biased 8-bit sequences, we got similar results: mutual correlations of MultiMCW, MultiMMC, and LZ78Y are very strong. Spearman's metric verified the same results. In addition to them, MCV was highly correlated with the same estimators MultiMCW, MultiMMC, and LZ78Y.

Table 3.19: Pearson correlation among different estimators for 8-bit biased distribution with 0.5 entropy

Pearson	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.1181	-0.0853	0.1748	-0.0292	0.1693	0.1693
t-Tuple	0.1181	1.0000	0.0038	0.1512	0.2076	0.1507	0.1508
LRS	-0.0853	0.0038	1.0000	-0.0316	0.0385	-0.0315	-0.0316
MultiMCW	0.1748	0.1512	-0.0316	1.0000	0.2497	1.0000	1.0000
Lag Prediction	-0.0292	0.2076	0.0385	**0.2497**	1.0000	0.2507	0.2508
MultiMMC	0.1693	0.1507	-0.0315	**1.0000**	**0.2507**	1.0000	1.0000
LZ78Y	0.1693	0.1508	-0.0316	**1.0000**	**0.2508**	**1.0000**	1.0000

Table 3.20: P-values of Pearson correlation among different estimators for 8-bit biased distribution with 0.5 entropy

P-values	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.1343	0.3040	0.0284	0.7104	0.0301	0.0301
t-Tuple	0.1343	0.0000	0.9574	0.0493	0.0074	0.0493	0.0493
LRS	0.3040	0.9574	0.0000	0.7104	0.7104	0.7104	0.7104
MultiMCW	0.0284	0.0493	0.7104	0.0000	0.0009	0.0000	0.0000
Lag Prediction	0.7104	0.0074	0.7104	**0.0009**	0.0000	0.0009	0.0009
MultiMMC	0.0301	0.0493	0.7104	**0.0000**	**0.0009**	0.0000	0.0000
LZ78Y	0.0301	0.0493	0.7104	**0.0000**	**0.0009**	**0.0000**	0.0000

Table 3.21: Spearman correlation among different estimators for 8-bit biased distribution with 0.5 entropy

Spearman	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.2834	0.0019	0.9511	0.0830	0.9506	0.9508
t-Tuple	**0.2834**	1.0000	0.1631	0.3001	0.0646	0.2931	0.2951
LRS	0.0019	0.1631	1.0000	-0.0006	0.0192	-0.0062	-0.0062
MultiMCW	**0.9511**	**0.3001**	-0.0006	1.0000	0.1037	0.9971	0.9972
Lag Prediction	0.0830	0.0646	0.0192	0.1037	1.0000	0.1091	0.1099
MultiMMC	**0.9506**	**0.2931**	-0.0062	**0.9971**	0.1091	1.0000	0.9999
LZ78Y	**0.9508**	**0.2951**	-0.0062	**0.9972**	0.1099	**0.9999**	1.0000

Table 3.22: P-values of Spearman correlation among different estimators for 8-bit biased distribution with 0.5 entropy

P-values	MCV	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.0001	0.9937	0.0000	0.3213	0.0000	0.0000
t-Tuple	**0.0001**	0.0000	0.0355	0.0000	0.4562	0.0000	0.0000
LRS	0.9937	0.0355	0.0000	0.9937	0.9413	0.9937	0.9937
MultiMCW	**0.0000**	**0.0000**	0.9937	0.0000	0.2015	0.0000	0.0000
Lag Prediction	0.3213	0.4562	0.9413	0.2015	0.0000	0.1842	0.1842
MultiMMC	**0.0000**	**0.0000**	0.9937	**0.0000**	0.1842	0.0000	0.0000
LZ78Y	**0.0000**	**0.0000**	0.9937	**0.0000**	0.1842	**0.0000**	0.0000

3.2.2.3 Correlation Analysis with Dataset 3: Non-IID-Low Entropy

Experiments were repeated with simulated biased datasets to measure the relations of the estimators when sequences do not satisfy the IID assumption and do have not full entropy. The following sequences were used for the experiments, similarly Pearson and Spearman coefficients were used to measure the correlation between entropy estimators. To make accurate observations Benjamini-Hochberg correction [5] applied to the P-values, $p < 0.01$ is assumed to be significant.

- **Non-IID binary distribution with entropy=** $H_{org} \times 0.875$**.** The dataset follows a biased binary distribution, where the elements of each sequence are generated as follows. Let $S = (s_1, s_2, s_3, \cdots)$ be a sequence of length $1\,000\,000$, all terms of the sequence are generated by the random number generator Mersenne Twister (MT19937) in `C++`, however for each k, $s_{8k} = \sum_{i=1}^{7} s_{8k-i} \mod 2$; that is, $8k^{th}$ element of the sequence is sum of previous seven elements in $\mod 2$. This modification reduces the entropy of the sequence in ratio $\frac{1}{8}$. The sequences in this dataset do not satisfy the IID-assumption. This dataset contains 200 binary sequences of length $1\,000\,000$.

The Pearson and Spearman correlations of the estimators are given in the following tables.

When we interpret Pearson correlation results of estimators for Non-IID biased binary sequences, it was observed a moderate correlation for (Markov, MCV) and (Markov, Collision).

Table 3.23: Pearson correlation among different estimators for Non-IID binary biased distribution with low entropy

Pearson	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.1302	0.5724	0.0125	-0.0266	-0.0739	-0.0027	-0.0423	0.0388	0.0914
Collision	0.1302	1.0000	0.3226	-0.0709	-0.1126	0.0301	0.0758	0.0433	0.0588	0.1270
Markov	**0.5724**	**0.3226**	1.0000	-0.0179	-0.0955	-0.0272	-0.0570	-0.0686	0.0669	0.0753
Compression	0.0125	-0.0709	-0.0179	1.0000	0.1121	0.0403	0.1025	-0.1645	0.0325	-0.2527
t-Tuple	-0.0266	-0.1126	-0.0955	0.1121	1.0000	-0.0661	0.0599	0.0920	-0.0497	-0.0597
LRS	-0.0739	0.0301	-0.0272	0.0403	-0.0661	1.0000	0.0388	-0.0042	0.0545	-0.0459
MultiMCW	-0.0027	0.0758	-0.0570	0.1025	0.0599	0.0388	1.0000	-0.0173	-0.0143	0.1074
Lag Prediction	-0.0423	0.0433	-0.0686	-0.1645	0.0920	-0.0042	-0.0173	1.0000	-0.0051	0.0636
MultiMMC	0.0388	0.0588	0.0669	0.0325	-0.0497	0.0545	-0.0143	-0.0051	1.0000	-0.0166
LZ78Y	0.0914	0.1270	0.0753	**-0.2527**	-0.0597	-0.0459	0.1074	0.0636	-0.0166	1.0000

Table 3.24: P-values of Pearson correlation among different estimators for non-iid binary biased distribution with low entropy

P-values	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.3306	0.0000	0.9152	0.8437	0.7040	0.9697	0.7706	0.7706	0.5500
Collision	0.3306	0.0000	0.0000	0.7040	0.4381	0.8398	0.7040	0.7706	0.7040	0.3319
Markov	**0.0000**	**0.0000**	0.0000	0.9063	0.5500	0.8437	0.7040	0.7040	0.7040	0.7040
Compression	0.9152	0.7040	0.9063	0.0000	0.4381	0.7706	0.4959	0.1106	0.8308	0.0019
t-Tuple	0.8437	0.4381	0.5500	0.4381	0.0000	0.7040	0.7040	0.5500	0.7572	0.7040
LRS	0.7040	0.8398	0.8437	0.7706	0.7040	0.0000	0.7706	0.9697	0.7148	0.7706
MultiMCW	0.9697	0.7040	0.7040	0.4959	0.7040	0.7706	0.0000	0.9063	0.9137	0.4648
Lag Prediction	0.7706	0.7706	0.7040	0.1106	0.5500	0.9697	0.9063	0.0000	0.9697	0.7040
MultiMMC	0.7706	0.7040	0.7040	0.8308	0.7572	0.7148	0.9137	0.9697	0.0000	0.9063
LZ78Y	0.5500	0.3319	0.7040	**0.0019**	0.7040	0.7706	0.4648	0.7040	0.9063	0.0000

According to Spearman's metric; similar to Pearson's metric, there was a moderate correlation for Markov and the MCV and Collision. Moreover, moderate correlations for the pairs (LZ78Y, MCV) and (LZ78Y, Markov) were observed.

Table 3.25: Spearman correlation among different estimators for non-iid binary biased distribution with low entropy

Spearman	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	1.0000	0.1140	0.5207	-0.0430	-0.0384	-0.0699	-0.0408	-0.0129	0.1048	0.4727
Collision	0.1140	1.0000	0.2729	-0.0454	-0.0762	0.0617	0.1123	-0.0371	0.0660	0.0628
Markov	**0.5207**	**0.2729**	1.0000	-0.0465	-0.0938	-0.0354	-0.0291	-0.0590	0.0464	0.6870
Compression	-0.0430	-0.0454	-0.0465	1.0000	0.1129	0.1117	0.0356	0.0456	0.0299	-0.0124
t-Tuple	-0.0384	-0.0762	-0.0938	0.1129	1.0000	-0.0001	0.0929	0.0534	0.0578	-0.1132
LRS	-0.0699	0.0617	-0.0354	0.1117	-0.0001	1.0000	-0.0109	-0.0237	0.0534	-0.0833
MultiMCW	-0.0408	0.1123	-0.0291	0.0356	0.0929	-0.0109	1.0000	-0.0252	0.0505	0.1024
Lag Prediction	-0.0129	-0.0371	-0.0590	0.0456	0.0534	-0.0237	-0.0252	1.0000	0.0036	-0.0633
MultiMMC	0.1048	0.0660	0.0464	0.0299	0.0578	0.0534	0.0505	0.0036	1.0000	0.0247
LZ78Y	**0.4727**	0.0628	**0.6870**	-0.0124	-0.1132	-0.0833	0.1024	-0.0633	0.0247	1.0000

Table 3.26: P-values of Spearman correlation among different estimators for non-iid binary biased distribution with low entropy

P-values	MCV	Collision	Markov	Compression	t-Tuple	LRS	MultiMCW	Lag Prediction	MultiMMC	LZ78Y
MCV	0.0000	0.4121	0.0000	0.7738	0.7738	0.7696	0.7738	0.9145	0.4656	0.0000
Collision	0.4121	0.0000	0.0005	0.7696	0.7087	0.7696	0.4121	0.7738	0.7696	0.7696
Markov	**0.0000**	**0.0005**	0.0000	0.7696	0.5302	0.7738	0.8122	0.7696	0.7696	0.0000
Compression	0.7738	0.7696	0.7696	0.0000	0.4121	0.4121	0.7738	0.7696	0.8122	0.9145
t-Tuple	0.7738	0.7087	0.5302	0.4121	0.0000	0.9985	0.5302	0.7696	0.7696	0.4121
LRS	0.7696	0.7696	0.7738	0.4121	0.9985	0.0000	0.9145	0.8208	0.7696	0.6345
MultiMCW	0.7738	0.4121	0.8122	0.7738	0.5302	0.9145	0.0000	0.8208	0.7696	0.4659
Lag Prediction	0.9145	0.7738	0.7696	0.7696	0.7696	0.8208	0.8208	0.0000	0.9791	0.7696
MultiMMC	0.4656	0.7696	0.7696	0.8122	0.7696	0.7696	0.7696	0.9791	0.0000	0.8208
LZ78Y	**0.0000**	0.7696	**0.0000**	0.9145	0.4121	0.6345	0.4659	0.7696	0.8208	0.0000

3.2.3 Impact of the Transformations

For this experiment, 200 uniformly distributed sequences of length $1\,000\,000$ with full entropy were used. These sequences were transformed using a reversing, binary derivative, and t-rotation for $t = 16, 64, 128, 1024$. Entropy estimates for the original and transformed sequences were compared, and their Pearson and Spearman correlation coefficients are listed in Table 3.27 and Table 3.28, respectively. Reversing and rotating the input sequences did not have any impact on its entropy estimation for the MCV, collision, Markov, t-tuple, and LRS estimators (hence, the same estimate is obtained) for either of the correlation metrics. Among different transformations, binary derivative seems to have the highest impact on the prediction-based estimates, namely multiMCW, Lag, multiMMC, and LZ78Y.

Table 3.27: Pearson Correlation according to the estimation results of transformed sequences

	Original	Reversed	Bin. Drv.	16-rot.	64-rot.	128-rot.	1024-rot.
MCV	1.0000	1.0000	-0.0289	1.0000	1.0000	1.0000	1.0000
Collision	1.0000	1.0000	-0.0160	1.0000	1.0000	1.0000	1.0000
Markov	1.0000	1.0000	0.4586	1.0000	1.0000	1.0000	1.0000
Compression	1.0000	0.3334	0.4887	0.3379	0.3374	0.3927	0.3368
t-Tuple	1.0000	1.0000	0.1144	1.0000	1.0000	1.0000	1.0000
LRS	1.0000	1.0000	0.7013	1.0000	1.0000	1.0000	1.0000
Multi MCW	1.0000	0.1301	0.8455	0.9999	0.9998	0.9997	0.9994
Lag Prediction	1.0000	0.1492	0.0037	0.9983	0.9971	0.9962	0.9915
MultiMMC	1.0000	0.0564	-0.0189	0.9977	0.9962	0.9962	0.8329
LZ78Y	1.0000	0.0598	0.1510	0.9961	0.9927	0.9918	0.9738

Table 3.28: Spearman Correlation according to the estimation results of transformed sequences

	Original	Reversed	Bin. Drv.	16-rot.	64-rot.	128-rot.	1024-rot.
MCV	1.0000	1.0000	-0.0432	1.0000	1.0000	1.0000	1.0000
Collision	1.0000	1.0000	0.0565	1.0000	1.0000	1.0000	1.0000
Markov	1.0000	1.0000	0.4030	1.0000	1.0000	1.0000	1.0000
Compression	1.0000	0.3090	0.5283	0.3053	0.3053	0.3685	0.3094
t-Tuple	1.0000	1.0000	0.0964	1.0000	1.0000	1.0000	1.0000
LRS	1.0000	1.0000	0.5425	1.0000	1.0000	1.0000	1.0000
Multi MCW	1.0000	0.8795	0.0170	0.9975	0.9954	0.9947	0.9869
Lag Prediction	1.0000	0.3607	-0.0282	0.9822	0.9717	0.9603	0.9219
MultiMMC	1.0000	0.3762	0.2872	0.9162	0.8772	0.8770	0.6943
LZ78Y	1.0000	0.6069	0.3580	0.9941	0.9884	0.9867	0.9530

3.2.4 Impact of the IID-Assumption Tests

For these experiments, the following datasets are used.

1. **Uniform distribution with full entropy.** The datasets are generated using the Cipher Block Chaining (CBC) mode of the block cipher Advanced Encryption Standard (AES) [9]. 200 binary sequences of length $1\,000\,000$ were generated. In these sequences, all outputs are assumed to have an equal probability of occurring, and are independent. Hence, the outputs have full entropy.

2. **Biased binary distribution with entropy=0.5.** The dataset follows a biased binary distribution, where the probability of observing a 1 is 0.7, and the probability of observing a 0 is 0.3. For each sample size, 200 sequences of length

$1\,000\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator Mersenne Twister (MT19937) in `C++`.

3. **4-bit near-uniform with entropy=0.5.** This dataset follows a 4-bit near- uniform distribution, where the probability of observing the template `0000` is 0.25, and the probability of observing other 4-bit templates is 0.05. For each sample size, 200 sequences of length $250\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

4. **8-bit near-uniform with entropy=0.5.** This dataset follows an 8-bit near-uniform distribution, where the probability of observing the template `00000000` is 0.06, and the probability of observing other 8-bit templates is 0.003686. For each sample size, 200 sequences of length $125\,000$ were generated. In these sequences, the expected entropy of a sequence is 0.5 per bit. This data is generated using the random number generator in `C++`.

5. **Non-IID binary distribution with entropy= $H_{org} \times 0.875$.** The dataset follows a biased binary distribution, where the elements of each sequence are generated as follows. Let $S = (s_1, s_2, s_3, \cdots)$ be a sequence of length $1\,000\,000$, all terms of the sequence are generated by the random number generator Mersenne Twister (MT19937) in `C++`, however for each k, $s_{8k} = \sum_{i=1}^{7} s_{8k-i} \mod 2$; that is, $8k^{th}$ element of the sequence is sum of previous seven elements in $\mod 2$. This modification reduces the entropy of the sequence in ratio $\frac{1}{8}$. The sequences in this dataset do not satisfy the IID-assumption. This dataset contains 200 binary sequences of length $1\,000\,000$.

For each dataset, IID-Assumption Tests and entropy estimators are employed. IID-Assumption Test results (number of passes and number of fails) are listed in the following table.

Table 3.29: IID-Assumption Test Results

	AES CBC		Biased Binary		Biased 4-bit		Biased 8-bit		Non-IID Binary	
	Pass	**Fail**	**Pass**	**Fail**	**Pass**	**Fail**	**Pass**	**Fail**	**Pass**	**Fail**
Excursion Test Statistic	200	0	200	0	199	1	200	0	200	0
Number of Directional Runs	200	0	200	0	200	0	200	0	200	0
Length of Directional Runs	200	0	200	0	200	0	200	0	200	0
Number of Increases and Decreases	199	1	200	0	200	0	200	0	200	0
Number of Runs based on the Median	200	0	200	0	198	2	199	1	200	0
Length of Runs based on Median	200	0	199	1	200	0	200	0	200	0
Average Collision Test Statistic	200	0	200	0	200	0	199	1	200	0
Maximum Collision Test Statistic	200	0	200	0	200	0	200	0	200	0
Periodicity Test Statistic (1)	200	0	200	0	200	0	200	0	200	0
Periodicity Test Statistic (2)	200	0	200	0	199	1	200	0	200	0
Periodicity Test Statistic (8)	200	0	200	0	200	0	200	0	200	0
Periodicity Test Statistic (16)	199	1	199	1	200	0	199	1	200	0
Periodicity Test Statistic (32)	200	0	200	0	200	0	200	0	200	0
Covariance Test Statistic (1)	200	0	200	0	200	0	199	1	200	0
Covariance Test Statistic (2)	200	0	200	0	200	0	200	0	199	1
Covariance Test Statistic (8)	200	0	200	0	200	0	199	1	199	1
Covariance Test Statistic (16)	200	0	200	0	200	0	200	0	200	0
Covariance Test Statistic (32)	200	0	200	0	200	0	200	0	200	0
Compression Test Statistic	200	0	199	1	198	2	199	1	0	200
-Square Goodness of Fit	200	0	200	0	200	0	200	0	200	0
-Square Independence	NA	NA	NA	NA	198	2	200	0	NA	NA
Length of the LRS Test	200	0	200	0	200	0	200	0	200	0

Almost all sequences of dataset *Uniform distribution with full* entropy generated by AES-CBC mode pass IID-Assumption tests, and entropy estimations are done by Most Common Value Estimate (IID-Track), as expected estimated entropy values are close to 1 and mean of all estimations is 0.9952.

On the other hand, the sequences of biased datasets *Biased binary distribution with entropy=0.5*, *4-bit near-uniform with entropy=0.5* and *8-bit near-uniform with entropy=0.5* again pass IID-Assumption tests, for each dataset, elements are generated identically and independent. Entropy estimations of these datasets again are done via Most Common Value Estimate (IID-Track); estimations are close to the real entropy value 0.5 per bit. The sequences of the dataset Non-IID binary distribution with entropy= $H_{org} \times 0.875$ cannot pass IID-Assumption tests, compression test statistic detects these sequences. Entropy estimation is done via Non-IID track. Table 3.30 shows the mean and standard deviations of the entropy estimation results.

Table 3.30: Entropy Estimation Results of IID sequences

	AES CBC		Biased Binary (H=0.5)		Biased 4-bit (H=0.5)		Biased 8-bit (H=0.5)	
	Mean	Std.Dev.	Mean	Std.Dev.	Mean	Std.Dev.	Mean	Std.Dev.
MCV	0.9952	0.0008	0.5121	0.0009	1.9935	0.0026	4.0442	0.0059
Collision	0.9107	0.0172	0.5095	0.0020	*	*	*	*
Markov	0.9982	0.0012	0.5145	0.0011	*	*	*	*
Compression	0.8496	0.0247	0.3224	0.0009	*	*	*	*
t-Tuple	0.9282	0.0107	0.5038	0.0103	1.9783	0.0171	4.0320	0.0161
LRS	0.9825	0.0195	0.7701	0.0178	3.2575	0.0746	7.0289	0.1735
MultiMCW	0.9961	0.0016	0.5120	0.0052	1.9925	0.0131	4.0363	0.0493
Lag Prediction	0.9960	0.0024	0.7765	0.0234	3.2927	0.0954	6.9225	0.3061
MultiMMC	0.9953	0.0088	0.5117	0.0052	1.9925	0.0131	4.0540	0.0527
LZ78Y	0.9960	0.0019	0.5117	0.0052	1.9925	0.0131	4.0540	0.0527

The sequences in the last dataset Non-IID binary distribution cannot pass the IID-Assumption tests, so the entropy estimation is done via Non-IID track. Because of generation method of the dataset for each k, $8k^{th}$ element of the sequence is the sum of previous seven elements in $\mod 2$, the expected entropy of the sequence is approximately $H_{original} \times 0.875$, assuming original entropy of the sequence generated by using the random number generator Mersenne Twister (MT19937) in `C++` is close to 1, then expected entropy estimation is close to 0.875. Collision estimate gives the minimum estimation for this data set, again we observe an underestimation because of Collision.

Table 3.31: Entropy Estimation Results of Non-IID sequences

	Non-iid (H=0.875)	
	Mean	Std.Dev.
MCV	0.9950	0.0009
Collision	0.8061	0.0073
Markov	0.9980	0.0013
Compression	0.8508	0.0248
t-Tuple	0.8807	0.0122
LRS	0.9477	0.0266
MultiMCW	0.9968	0.0171
Lag Prediction	0.9956	0.0064
MultiMMC	0.8278	0.0052
LZ78Y	0.9947	0.0084

3.3 Discussion

In this chapter of the thesis, we studied the black-box entropy estimators described in NIST SP 800-90B. We observed that compression and collision estimates both underestimate the entropy for uniform and biased distributions, which is consistent with the findings of Zhu et al. [35] and Kim et al. [17]. The remaining estimates are close to the true entropy for the uniform distribution. However, LRS and lag prediction overestimate entropy for binary, 4-bit, and 8-bit sequences for biased distributions. Understanding the reasons for this gap based on the details of the estimators is planned for future work.

These experiments show a strong correlation between the Markov and MCV tests for uniform distribution. For uniform distribution, experiments were repeated for sequences of different lengths. Even if a small decrease in the correlation coefficients was observed when the experiments were repeated, the general results of experiments are close to each other.

When correlation analysis was conducted with biased datasets satisfying IID- assumption, there was an increase in correlation amounts and also the number of correlated estimators. Experiments showed that mutual correlations of MultiMCW, MultiMMC, and LZ78Y are very strong. In addition to them, MCV was highly correlated with the estimators MCV and the estimators Markov, Compression, MultiMCW, MultiMMC, and LZ78Y. On the other hand, when we interpret correlation results of estimators for biased datasets not satisfying IID-Assumption, it was observed a moderate correlation between pairs (Markov, MCV) and (Markov, Collision). Moreover, moderate correlations for the pairs (LZ78Y, MCV) and (LZ78Y, Markov) were observed.

Additionally, we observed that taking binary derivation significantly changes the entropy estimates, especially for prediction-based estimators.

CHAPTER 4

ESTIMATING ENTROPY VIA INDEX-VALUE COINCIDENCE

In this chapter, a new entropy estimator called *index-value coincidence estimate* is introduced in detail. We provide mathematical background, some applications, experimental results, and comparisons to demonstrate the effectiveness of the estimator.

4.1 Index-Value Coincidence Estimator

This section provides the details of the theoretical background and the description of the *index-value coincidence* estimator. Let $S = (s_1, s_2, \dots, s_L)$ be a sequence generated by a TRNGs, where $s_i \in A = \{x_1, x_2, \dots, x_k\}$. The estimator assigns the values from the alphabet set A to the sequence S and checks the number of times the assignment is the same as the corresponding value of the sequence. The assignment is done as follows: $s_1 \leftrightarrow x_1, s_2 \leftrightarrow x_2, \dots, s_i \leftrightarrow x_i$ ans so on. If $s_i = x_i$ for any i, this is called *i-index-value coincidence* point. After an i-index-value coincidence point, labeling restarts from x_1. If there are no correct assignments for all k values, the endpoint is called *exhausting point* (EP). In that case, the assignment starts over from x_1. Index-value coincidence points and exhausting points of the sequence are called *terminal* points.

Example. Let $S = (3, 7, 6, 5, 2, 1, 4, 7, 3, 2, 2, 3, 3, 1, 8, 2)$ be a sequence of length 16 formed by the elements in $A = \{1, 2, 3, 4, 5, 6, 7, 8\}$. The index-value coincidence and exhaustive points of S are shown in the following table. As shown in Table 4.1, there is no coincidence in the values of s_i's and x_i's (i.e., $s_i \neq x_i$) for the first $k = 8$ terms, hence $i = 8$ is an exhausting point. Then, starting from $i = 9$, the labeling restarts from 1, and we see that s_{10} is a 2-index-value coincidence point. Similarly, labeling starts from 1 after each terminal point, s_{13} is a 3-index-value coincidence point and s_{16} is a 2-index-value coincidence point.

Table 4.1: Example demonstration of index-value coincidences

i	s_i	x_i	Index-coincidence	EP?	i	s_i	x_i	Index-coincidence	EP?
1	3	1	no	-	9	3	1	no	-
2	7	2	no	-	10	2	2	yes	-
3	6	3	no	-	11	2	1	no	-
4	5	4	no	-	12	3	2	no	-
5	2	5	no	-	13	3	3	yes	-
6	1	6	no	-	14	1	1	yes	-
7	4	7	no	-	15	8	1	no	-
8	7	8	no	yes	16	2	2	yes	-

4.1.1 Estimating the Probability Distribution

Let $\mathcal{X}$ be the random variable taking values from the set $A = \{x_1, x_2, \ldots, x_k\}$ with probabilities $P(\mathcal{X} = x_i) = p_i$, where p_i is the probability of $\mathcal{X}$ taking the value x_i. To compute the theoretical probabilities of index-value coincides and exhausting points, the following probabilities are defined:

Probability	**Definition**
$T(n)$	the probability that nth element of the input sequence S is a terminal point (i.e., index-value coincidence or an exhausting point)
$T_i(n)$	the probability that nth element of the input sequence S is an i-index-value coincidence point.
$T_{E.P.}(n)$	the probability that nth element of the input sequence S is an exhausting point.

The following equation provides a recurrence relation for $T(n)$:

$$\begin{aligned}
T(n) &= T_1(n) + T_2(n) + \cdots + T_k(n) + T_{E.P.}(n) \\
&= T(n-1)p_1 + T(n-2)(1-p_1)p_2 + \cdots \\
&\quad +T(n-k)(1-p_1)(1-p_2)\cdots(1-p_k-1)p_k + \\
&\quad +T(n-k)(1-p_1)(1-p_2)\cdots(1-p_{k-1})(1-p_k)
\end{aligned}$$

Note that this recursion assumes that sample values are independently selected. The characteristic equation of this recurrence relation is

$$\begin{aligned}
&\lambda^k - p_1\lambda^{k-1} - (1-p_1)p_2\lambda^{k-2} - (1-p_1)(1-p_2)p_3\lambda^{k-3} \\
&- \cdots - (1-p_1)(1-p_2)\cdots(1-p_{k-1})p_k \\
&-(1-p_1)(1-p_2)\cdots(1-p_{k-1})(1-p_k) = 0
\end{aligned}$$

which has k solutions such that $\lambda_1 = 1$ and for $i = 2, 3, \ldots, k$, $\lambda_i \in \mathbb{C}$, $|\lambda_i| < 1$. The general formula for $T(n)$ is

$$T(n) = C_1\lambda_1^n + C_2\lambda_2^n + C_3\lambda_3^n + \cdots + C_k\lambda_k^n$$

where $\lambda_1 = 1$ and for $i = 2, 3, \ldots, k$, $\lambda_i \in \mathbb{C}$, $|\lambda_i| < 1$ $C_i \in \mathbb{R}$. Since for $i = 2, 3, \ldots, k$ we have $|\lambda_i| < 1$ for sufficiently large n, we have

$$\lim_{n\to\infty} T(n) = \lim_{n\to\infty}\left(C_1 + C_2\lambda_2^n + C_3\lambda_3^n + \cdots + C_k\lambda_k^n\right) = C_1$$

Given the sequence $S = (s_1, s_2, \ldots, s_L)$ where $s_i \in A = \{x_1, x_2, \ldots, x_k\}$ for each $i \in \{1, 2, \ldots, k\}$. Let t_i be the number of index-value coincidence for each x_i, and $t_{E.P.} = t_0$ be the number of exhausting points where there is no index-value coincidence over the entire alphabet set $\{x_1, x_2, \ldots, x_k\}$ correspondence with the sequence. The number of total probable situations can be evaluated with

$$t = \sum_{i=0}^{k} t_i\ .$$

Following,

$$\begin{aligned}
T(n) &= C_1 + C_2\lambda_2^n + C_3\lambda_3^n + \cdots + C_k\lambda_k^n \\
&\approx C_1 \\
&\approx \frac{t_0}{t} + \frac{t_1}{t} + \cdots + \frac{t_k}{t},
\end{aligned}$$

the expected values of t_i's i.e., the number of index-value coincidence for each x_i can be evaluated as follows:

$$\begin{aligned} E(t_1) &= tp_1 && (4.1)\\ E(t_2) &= t(1-p_1)p_2 && (4.2)\\ E(t_k) &= t(1-p_1)(1-p_2)\cdots(1-p_{k-1})p_k && (4.3)\end{aligned}$$

These equations allow us to compute the entire probability distribution $P = \{p_1, p_2, \ldots, p_k\}$ of the set $A = \{x_1, x_2, \ldots, x_k\}$ as

$$\begin{aligned} p_1 &= \frac{t_1}{t} && (4.4)\\ p_2 &= \frac{t_2}{t(1-p_1)} && (4.5)\\ &\ldots && (4.6)\\ p_k &= \frac{t_k}{t(1-p_1)\cdots(1-p_{k-1})} && (4.7)\end{aligned}$$

4.1.2 Description of the Index-Value Coincidence Estimator

For a given sequence $S = (s_1, s_2, \ldots, s_L)$, where $s_i \in A = \{x_1, x_2, \ldots, x_k\}$, the test procedure can be summarized as follows:

1. Determine the number of i-index-value coincidence, denoted $O(t_i)$, and exhausting points, denoted $O(t_{E.P.}) = O(t_0)$.
2. By using observed values of t_i s and $t_{E.P.} = t_0$ compute the estimated probability distribution $Q = \{q_1, q_2, \ldots, q_k\}$ of the elements of $A = \{x_1, x_2, \ldots, x_k\}$.

$$t = \sum_{i=0}^{k} O(t_i)$$

For any i, $O(t_i) \neq t$, and $O(t_0) \neq t$

$$\begin{aligned} O(t_1) &= tq_1 \Longrightarrow q_1 = \frac{O(t_1)}{t}\\ O(t_2) &= t(1-q_1)p_2 \Longrightarrow q_2 = \frac{O(t_2)}{t(1-q_1)}\\ &\vdots\\ O(t_k) &= t(1-q_1)(1-q_2)\cdots(1-q_{k-1})q_k \Longrightarrow q_k = \frac{O(t_k)}{t(1-q_1)\cdots(1-q_{k-1})}\end{aligned}$$

3. For each i, evaluate $p_i = \dfrac{q_i}{q_1 + q_2 + \cdots + q_k}$.
4. Evaluate the min-entropy of the generator as $H_\infty = -\log(\max_i p_i)$.

Since this method gives estimated probabilities of all elements of the alphabet set, it can be used to estimate different types of entropy measures. In this study, we focused on the min-entropy as also done in SP 800-90B.

Example 4.1.1. *Let* $S = (0110010100100011)$ *be a binary sequence of length 16 formed by the elements in* $A = \{0, 1\}$*. Then direct assignment in* A *to the elements of the sequence gives the index-value coincidences as follows:*

Table 4.3: Example index-value coincidences values for a binary sequence

Sequence	0	1	1	0	0	1	0	1	0	0	1	0	0	0	1	1
Index	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1

The observed values of index-value coincidences and exhausting points are

$$O(t_0) = 6, O(t_1) = 2, O(t_{E.P.}) = 3.$$

The total number of terminal points is $t = O(t_{E.P.}) + O(t_0) + O(t_1) = 3 + 6 + 6 = 11$ *then*

$$q_0 = \frac{O(t_0)}{t} = \frac{6}{11} = 0.5454$$

and

$$q_1 = \frac{O(t_1)}{t(1 - q_1)} = \frac{2}{11(1 - 0.5454)} = 0.3999$$

. Estimated probabilities are evaluated as

$$p_0 = \frac{0.5454}{0.5454 + 0.3999} = 0.5770$$

$$p_1 = \frac{0.3999}{0.5454 + 0.3999} = 0.4230$$

Estimated probability distribution of the set $A = \{0, 1\}$, can be calculated as $P = \{p_1 = 0.5770, p_2 = 0.4230\}$ and the min-entropy estimation is

$$H_{min} = -\log(0.5770) = 0.7935$$

and the Shannon entropy is

$$H = -0.5770 \log(0.5770) - 0.4230 \log(0.42300) = 0.9829$$

.

4.2 Experimental Results

For the following experiments estimators specified in NIST SP 800-90B and the index-value coincidence estimator are applied to the following simulated data sets:

Uniform distribution with full entropy. The dataset contains 100 sequences of length $1\,000\,000$, each sequence is generated by using Cipher Block Chaining (CBC) mode of the block cipher Advanced Encryption Standard (AES) [9]. Required input sequences are generated by using the generators "std: :random_device" and "std: :mt19937" of the standard library "random" in C++. The outputs are assumed to have full entropy. (This dataset is tested as in original binary representation and in 4-bit representation.)

Biased binary distribution with entropy=0.5. The dataset contains 100 sequences of length $1\,000\,000$, and each sequence has a biased binary distribution with the probabilities of observing a 1 is 0.7 and observing a 0 is 0.3. The expected entropy of a sequence is evaluated as 0.5 per bit.

4-bit near-uniform with entropy=0.5. The dataset contains 100 sequences of length $1\,000\,000$, each sequence has a 4-bit near-uniform distribution with the probabilities of observing the template `0000` is 0.25, and observing other 4-bit templates is 0.05. The expected entropy of a sequence is evaluated as 0.5 per bit.

8-bit near-uniform with entropy=0.5. The dataset contains 100 sequences of length $1\,000\,000$, each sequence has an 8-bit near-uniform distribution with the probabilities of observing the template `00000000` is 0.06, observing other 8-bit templates is 0.003686. The expected entropy of a sequence is evaluated as 0.5 per bit.

4.2.1 Comparison for *min-entropy*: Index-Value vs. NIST SP 800-90B Estimators

To measure the accuracy of index-value coincidence estimator, entropy estimations are calculated for all simulated datasets and *min-entropy* estimations compared with the estimators in NIST SP 800-90B. Results are given in the following tables:

Table 4.4: Mean and standard deviation of min-entropy estimators for uniform distributions: binary with full entropy, and 4-bit with full entropy sources

	Binary Full Entropy		**4-bit Full Entropy**		
	Mean	**Std. Dev.**	**Mean**	**Mean/Bit**	**Std. Dev.**
MCV	0.9951	0.0010	3.9520	0.9880	0.0050
Collision	0.9142	0.0194	*	*	*
Markov	0.9982	0.0011	*	*	*
Compression	0.8498	0.0277	*	*	*
t-Tuple	0.9278	0.0099	3.7819	0.9455	0.0145
LRS	0.9852	0.0163	3.8967	0.9742	0.0909
Multi MCW	0.9954	0.0056	3.9566	0.9892	0.0832
Lag Prediction	0.9961	0.0020	3.9663	0.9916	0.0494
Multi MMC	0.9953	0.0088	3.9466	0.9866	0.1229
LZ78Y	0.9956	0.0088	3.9467	0.9867	0.1229
Index-Value	0.9988	0.0010	3.9193	0.9798	0.0197

Table 4.5: Mean and standard deviation of min-entropy estimators for biased distributions: binary with 0.5 entropy, 4-bit with 0.5 entropy, and 8-bit with 0.5 entropy sources

	Binary 0.5 Entropy		**4-bit 0.5 Entropy**			**8-bit 0.5 Entropy**		
	Mean	**Std. Dev.**	**Mean**	**Mean/Bit**	**Std. Dev.**	**Mean**	**Mean/Bit**	**Std. Dev.**
MCV	0.5122	0.0009	2.0448	0.5112	0.0054	4.0737	0.5092	0.0184
Collision	0.5095	0.0021	*	*	*	*	*	*
Markov	0.5145	0.0012	*	*	*	*	*	*
Compression	0.3225	0.0009	*	*	*	*	*	*
t-Tuple	0.5038	0.0091	2.0240	0.5060	0.0262	4.0524	0.5065	0.0512
LRS	0.7718	0.0159	3.0712	0.7678	0.0836	6.1378	0.7672	0.1212
Multi MCW	0.5116	0.0073	2.0411	0.5103	0.0339	4.0657	0.5082	0.0873
Lag Prediction	0.7764	0.0244	3.1145	0.7786	0.0468	6.0944	0.7618	0.1504
Multi MMC	0.5113	0.0073	2.0408	0.5102	0.0338	4.1875	0.5234	0.1137
LZ78Y	0.5113	0.0073	2.0408	0.5102	0.0339	4.1915	0.5239	0.1147
Index-Value	0.5146	0.0009	2.0580	0.5145	0.0203	4.1836	0.5230	0.3356

The results of these experiments show that the index-value coincidence estimate gives accurate *min-entropy* estimates for simulated data sets.

4.2.2 Comparison for *Shannon-entropy*: Index-Value vs. NIST SP 800-90B Estimators

NIST SP 800-90B estimators give only min-entropy estimations for the noise source; on the other hand, the index-value coincidence estimator estimates the all-probability distribution, so this estimator gives min-entropy and also Shannon entropy estimates. To make a meaningful comparison about Shannon entropy estimations binary datasets are used for experiments.

NIST SP 800-90B estimators give estimation results as $H_{min} = -\log_2(\max_{1\leq i\leq n} p_i)$. For alphabet set $A = \{0, 1\}$, solving the equation $H_{min} = -\log_2(\max_{1\leq i\leq n} p_i)$ for p_{max} and calculating the probability of the second element of the alphabet set as $1 - p_{max}$, it is possible to calculate Shannon entropy estimations. (However, for an alphabet set containing more than two elements, it is not possible to calculate Shannon entropy for given inputs.)

To Compare Shannon entropy estimations of index-value coincidence estimator and NIST SP 800-90B estimators, datasets with uniform distribution with full entropy and biased binary distribution are used. Each dataset contains 100 binary sequence of length 1 million. For the dataset uniform distribution with full entropy, the probabilities of observing a 0 or 1 are equal and 0.5 for each sequence. The expected Shannon entropy is 1. For the dataset biased binary distribution, probability of observing a 1 is 0.7 and observing a 0 is 0.3. Shannon entropy is calculated as 0.881291.

Entropy estimations are calculated for simulated datasets and *Shannon-entropy* estimations are compared. Results are given in the following table:

Table 4.6: Mean and standard deviation of Shannon-entropy estimations for uniform binary distribution with full entropy, and biased binary distribution sources

	Binary Full Entropy		**Biased Binary**	
	Mean	**Std.Dev.**	**Mean**	**Std.Dev.**
MCV	0.999991	3.81E-06	0.879870	0.000564
Collision	0.997134	1.17E-03	0.878244	0.001257
Markov	0.999998	1.83E-06	0.881270	0.000691
Compression	0.990954	2.92E-03	0.722566	0.001017
t-Tuple	0.998061	5.29E-04	0.874711	0.005719
LRS	0.999827	3.85E-04	0.978573	0.003604
Multi MCW	0.999981	1.10E-04	0.879462	0.004949
Lag Prediction	0.999993	1.07E-05	0.979278	0.006603
Multi MMC	0.999964	3.01E-04	0.879304	0.004931
LZ78Y	0.999965	3.01E-04	0.879304	0.004931
Index-Value	**0.999999**	1.17E-06	**0.881304**	0.000556
Expected Ent	**1.000000**		**0.881291**	

The results of these experiments show that the index-value coincidence estimate gives more accurate *Shannon-entropy* estimates compared to NIST SP 800-90B estimators. For these datasets, Most Common Value and Markov estimate results are close to the expected Shannon entropy estimation ın the test suite.

4.2.3 Correlation of Estimators

To measure the mutual correlation of the index-value coincidence estimator with estimators of NIST SP 800-90B Pearson and Spearman correlation coefficients are calculated for the dataset uniform binary distribution with full entropy.

Table 4.7: Pearson correlation among index-value coincidence and NIST SP 800-90B estimators for uniform binary distribution with full entropy

	MCV	**Collis.**	**Markov**	**Compre.**	**t-Tuple**	**LRS**	**MMCW**	**Lag**	**MMMC**	**LZ78Y**	**Ind-Val**
MCV	1.0000	0.0471	0.5167	-0.0976	-0.0517	-0.0501	-0.0350	0.0449	0.0395	0.3650	**0.4502**
Collis.		1.0000	0.2706	0.0375	-0.0667	0.0632	0.0744	0.0457	0.1012	-0.1231	0.2211
Markov			1.0000	-0.0760	-0.0975	0.0677	-0.1014	0.0216	-0.0269	0.4700	**0.6300**
Compre.				1.0000	-0.0040	-0.0496	0.1961	0.0777	-0.0077	-0.0266	-0.0925
t-Tuple					1.0000	-0.0149	-0.0053	-0.0024	0.1870	-0.0121	-0.0035
LRS						1.0000	-0.0389	0.1072	-0.0546	-0.0278	-0.0871
MMCW							1.0000	0.0223	-0.2226	-0.1082	-0.0473
Lag								1.0000	0.0435	-0.0257	-0.0966
MMMC									1.0000	0.0084	0.0342
LZ78Y										1.0000	0.2502
Ind-Val											1.0000

Table 4.8: Spearman correlation among index-value coincidence and NIST SP 800-90B estimators for uniform binary distribution with full entropy

	MCV	**Collision**	**Markov**	**Compre.**	**t-Tuple**	**LRS**	**MMCW**	**Lag**	**MMMC**	**LZ78Y**	**Ind-Val**
MCV	1.0000	0.0900	0.5248	-0.1229	-0.0919	-0.0156	-0.0564	-0.0179	0.1291	0.5149	**0.3689**
Collis.		1.0000	0.2396	0.0414	-0.0604	-0.0230	0.0761	0.0858	-0.0479	-0.0576	0.3020
Markov			1.0000	-0.0771	-0.0995	0.0171	-0.1282	0.0519	0.1798	0.7746	**0.4817**
Compre.				1.0000	0.0406	-0.1212	0.1527	0.0167	-0.0687	-0.0228	-0.0306
t-Tuple					1.0000	0.0790	-0.0108	-0.0313	0.0558	-0.0934	-0.0264
LRS						1.0000	-0.0167	-0.1029	-0.0331	-0.0609	-0.0589
MMCW							1.0000	-0.0340	-0.3049	-0.1692	0.0252
Lag								1.0000	0.0439	0.0270	-0.1206
MMMC									1.0000	0.2775	0.1876
LZ78Y										1.0000	0.3142
Ind-Val											1.0000

According to Pearson and Spearman correlations, there is a moderate correlation between (Index-Value, Markov). Also, it is observed that mutual correlations of index-value and most common value estimates can be accepted as moderate according to Pearson's metric. Mutual correlation of index-value coincidence with other estimatios are low and can be ignored.

Experiments and correlation analysis results show that index-value coincidence is a good alternative for estimating the entropy of non-binary sequences instead of the Markov estimate, which is suitable for just binary inputs.

We focus on the black-box statistical entropy estimation of TRNGs and propose an index-value coincidence estimate with its statistical and mathematical background. Experimental results show that index-value coincidence estimate gives expected and accurate entropy estimations for simulated datasets. We provided some experimental comparisons and correlation analysis with the estimators specified in NIST SP 800-90B, and results show that index-value coincidence estimates may be included in the existing estimators to increase the number of estimators and the accuracy of results. Moreover, contrary to existing estimators, index-value coincidence estimate gives *Shannon-entropy*, similarly Shannon entropy estimates of index-value coincidence and NIST SP 800-90B estimators are compared, and it is clearly observed that index-value coincidence give more accurate estimations for Shannon entropy.

CHAPTER 5

HEALTH TESTS FOR CRYPTOGRAPHIC RANDOM NUMBER GENERATORS

In this chapter, existing health test suites are examined and a health test suite for cryptographic TRNGs is introduced by using random variables *weight, run, runs of length 1* and *overlapping templates*, some suggested parameters and experimental results are given.

True random number generators are hardware mechanisms that generate random number sequences from some physical events. In the operating process of mechanical devices, the process may be affected by outside conditions such as humidity, temperature, pressure, etc. These effects also may change the conditions of the physical events that are employed as noise sources of the device. To detect unexpected deviations in the working process of a TRNG, health tests are designed as components of their structures.

Health tests are designed to detect corruption and error in the working mechanism of the entropy source and give warnings about disruptions in the process, simultaneously. For this reason, tests should be designed as algorithms that provide fast results and have low time complexity. NIST SP 800-90B[32] and FIPS PUB 140-2 [19] recommend some health tests and describe testing process. Based on statistical randomness tests found in the literature, it is possible to define health tests to be used for these purposes. In this study, by using random variables *weight, run, runs of length 1* and *overlapping templates* we introduce a health test suite for cryptographic random number generators.

5.1 Details of Existing Health Test Suites

There are some standards and guidelines for randomness, they provide designing principles of a RNG, statistical randomness test, entropy estimation methods and health tests. NIST SP 800-90B[32] and FIPS PUB 140-2 [19] describe health tests for RNGs.

NIST SP 800-90B *Recommendation for the Entropy Sources Used for Random Bit Generation*[32] gives a guideline for requirements of health tests and provides two approved health tests Repetition Count test, and the Adaptive Proportion test.

1. **Repetition Count Test:** This test detects failures that cause the noise source to generate the same output value for a long period of time.
2. **Adaptive Proportion Test:** This test detects a large loss of entropy that might occur as a result of some physical or environmental changes. Adaptive proportion test determines if the sample occurs too frequently by measuring the frequency of a sample value in a sequence of noise source outputs.

FIPS PUB 140-2 *Security Requirements for Cryptographic Modules*[19] describes statistical random number generator tests, for a cryptographic module employing RNG. Consecutive $20,000$ bits of output of RNG are tested by the following tests:

1. **Monobit Test:** X is defined as the weight of the sequence of length $20,000$. If $9,725 < X < 10,275$ holds, then the test is passed.
2. **The Poker Test:** The sequence of length $20,000$ is divided into 4-bit non-overlapping subsequences. The number of all possible 4-bit templates is 16. For each 4-bit template, let $f(i)$ denote the number of the i template in the sequence for $0 \leq i \leq 15$. Evaluate

$$X = \frac{16}{5000}(\sum_{i=0}^{15}[f(i)]^2) - 5000$$

If $2.16 < X < 46.17$ holds, the test is passed.

3. **The Runs Test:** A run is a maximal sequence of consecutive identical bits. For this tests, numbers of runs of the sequence of length $20,000$ are counted and stored according to their lengths. The test is passed, if the numbers of runs are in the required intervals:

Length of Run	Required Interval
1	2,343-2,657
2	1,135-1,365
3	542-708
4	251-373
5	111-201
6+	111-201

4. **The Long Run Test:** A run of length 26 or more is defined as long run. The test is passed if there is no long run of sequence.

These tests are examined, and to increase the number of tests in the health test suite, mathematical and statistical backgrounds of tests and random variables are analyzed in detail.

5.2 Proposed Health Tests

By preserving similar evaluation frameworks with existing methods, a basic mathematical model of a health test suite is constructed with some selected random variables. The health test suite is created to ensure that the entropy of the random bit generator remains below the expected level during the operation and to detect problems and faulty (mechanical or software) processes that may occur in the mechanism. The basic evaluation principle of the defined health test suite is to detect the subsequences that are at the extreme limits in the distribution functions of the selected random variables and to keep them under control.

Motivation: Let Ω_n be the set of all binary sequences of length n, and X be a random variable, defined as $X : \Omega_n \to \mathbb{T}$ where $\mathbb{T}$ is a finite subset of non-negative integers.

Assume that the probability distribution function of X is as follows.

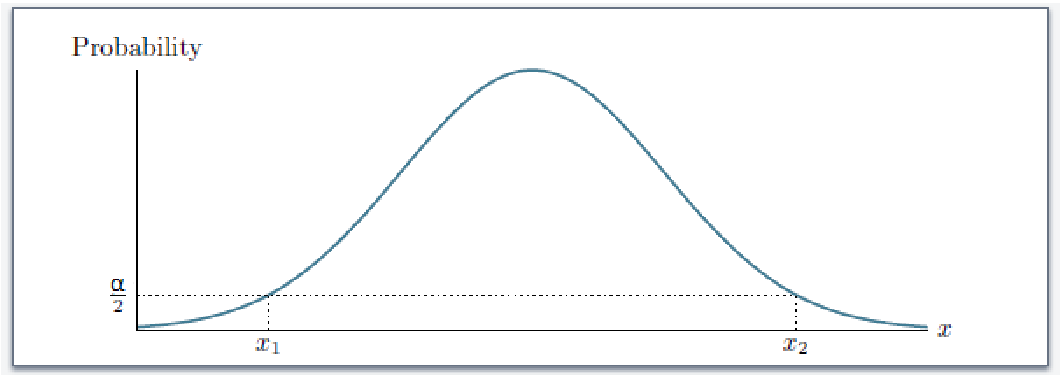

Figure 5.1: Probability Distribution of Random Variable X

For a binary sequence $S \in \Omega_n$, let $X(S) = k$. For a specified significant level α, if $Pr(X = k) \in (0, \frac{\alpha}{2})$ or $Pr(X = k) \in (1 - \frac{\alpha}{2}, 1)$; that is, $k \in I = (x_1, x_2)$, the sequence S is considered at the extreme limits in the probability distribution function of the random variable X.

This test suite is designed to detect such subsequences of the output of RNG, at the extreme limits in the probability distribution functions of the random variables *weight, run, runs of length 1* and *overlapping templates*. For each random variable, according to their probability distribution functions and the significance level of the statistical test suite, valid intervals are determined.

5.2.1 Test Description

Some definitions and parameters are given as:
S: Binary sequence
α: Significance level (False positive probability – the probability that a correctly functioning noise source will fail the test on a given output.)
W: Window size (Length of each subsequence)
k: Repetition number for each test
T_i: Random variable
I_i: Valid interval for the random variable T_i

Binary sequence S is divided into k non-overlapping subsequences of length W-bit. If the length of the sequence is greater than $k.W$, the remaining terms will be omitted.

This test suite is constructed by seven random variables, to preserve the significance level α for the total evaluation, total significance levels α_i of each random variable can be evaluated by the following formula:

$$\alpha = 1 - (1 - \alpha_i)^7$$

Since each random variable is employed k-times, to test subsequences of S, for each repeated test significance levels β can be evaluated by the following formula:

$$\alpha_i = 1 - (1 - \beta)^k$$

β determines the valid intervals I_i s for each test. According to probability distribution functions of random variables, for each random variable valid intervals are determined. (This part contains some detailed calculation processes.)

The sequence S is deemed to pass test i, if for each subsequence of S, T_i is in the valid interval I_i. The algorithm of the test suite is as follows:

Algorithm 1 Health Test Suite

INPUT: A binary sequence S

```
1: Divide $S$ into subsequences $s_i$ s
2: for $i = 1$ to $7$ do
3:     for $j = 1$ to $k$ do
4:         if $T_i(s_j) \notin I_i$, "test $i$ failed" then
5:         end if
6:     end for
7: end for
8: “pass"
```

5.2.2 Random Variables

In this section, definitions, probability distribution functions, and some useful recursions of selected random variables used in the health test suite are given. For definitions and mathematical details of random variables [18] is used, more details of the random variables can be seen in [18].
Let Ω_n be the set of all binary sequences of length n. For a binary sequence $S = (s_1, s_2, \cdots, s_n)$ of length n, where $s_i \in \{0, 1\}$ for each i, a random variable is defined as $T : \Omega_n \to \mathbb{T}$ where $\mathbb{T}$ is a finite subset of non-negative integers.

5.2.2.1 Weight

Weight is defined as the number of 1's in the given sequence S:

$$T(S) = \sum_{i=1}^{n} s_i$$

For example; weight of the sequence $S = (0111001010011010)$ is 9.
Probability Distribution Function:

$$F_n(k) = 2^{-n} \sum_{i=0}^{k} \binom{n}{i}$$

Useful Recursions: Initial values:$F_1(0) = 1, F_1(1) = \frac{1}{2}$
For $n \geq 2$ and $k = 0$

$$F_n(0) = \frac{1}{2} F_{n-1}(0)$$

For $n \geq 2$ and $k \geq 1$

$$F_n(k) = \frac{1}{2}[F_{n-1}(k) + F_{n-1}(k-1)]$$

Whenever $k \geq n$, $F_n(k) = 1$.

5.2.2.2 Number of Total Runs

A run is defined as consecutive identical bits of the sequence. The number of total runs counts the runs of the given sequence S:

$$T(S) = \text{Number of total runs of } S$$

For example; the sequence $S = (0111001010011010)$ has 11 runs: 0, 111, 00, 1, 0, 1, 00, 11, 0, 1 and 0.

Probability Distribution Function:

$$F_n(k) = 2^{-n+1} \sum_{i=0}^{k} \binom{n-1}{i-1}$$

Useful Recursions: Initial values:$F_1(0) = 0, F_1(1) = 1$

For $n \geq 2$ and $k = 1$

$$F_n(0) = \frac{1}{2} F_{n-1}(0)$$

For $n \geq 2$ and $k \geq 2$

$$F_n(k) = \frac{1}{2}[F_{n-1}(k) + F_{n-1}(k-1)]$$

Whenever $k \geq n$, $F_n(k) = 1$.

5.2.2.3 Number of Runs of Length-1

Number of runs of length-1 counts the number of lengths of 1 runs of the given sequence.

$$T(S) = \text{Number of runs of Length-}1 S$$

For example; the sequence $S = (0111001010011010)$ has 11 runs: 0, 111, 00, 1, 0, 1, 00, 11, 0, 1 and 0 so the number of runs of length-1 is 7.

Probability Distribution Function: Let $C_1(n, k)$ be the number of sequences of length n, containing k runs of length-1.

$$F_n(k) = 2^{-n+1} \big(\sum_{i=1}^{n} C_1(n-i, k) - C_1(n-1, k) + C_1(n-1, k-1) \big)$$

Useful Recursions:

$$C_1(n, k) = 2C_1(n-1, k) - C_1(n-1, k) + C_1(n-2, k) +$$

$$C_1(n-1, k-1) - C_1(n-2, k-1)$$

5.2.2.4 Overlapping Templates

This random variable counts the frequency of a predefined template of length l in the overlapping l-bit divisions of the given sequence.

$$T(S) = \text{Frequency of a predefined template in } S$$

For example; in the sequence $S = (0111001010011010)$ the template 11 of length-2 is seen 3 times.

Probability Distribution Functions: For this test suite templates of length 4 are used, according to the overlapping structure of the templates their probability distribution functions are given as follows:

Let $T(n, k)$ be the number of sequences of length n with k overlapping substrings of length 4.

0-overlapping templates: 0001, 0011, 0111, 1000, 1100, 1110.

$$T(n, 0) = 2T(n-1, 0) - T(n-4, 0)$$

$$T(n,k) = \begin{cases} 0 & \text{if } n < 4k \\ 1 & \text{if } n = 4k \\ 2T(n-1,k) - T(n-4,k) + T(n-4,k-1) & \text{if } n > 4k \end{cases}$$

1-overlapping templates: 0010, 0100, 1011, 1101, 0110, 1100.

$$T(n, 0) = 2T(n-1, 0) - T(n-3, 0) + T(n-4, 0)$$

$$T(n,k) = \begin{cases} 0 & \text{if } n < 3k+1 \\ 1 & \text{if } n = 3k+1 \\ 2T(n-1,k) - T(n-3,k) + & \\ T(n-4,k) + T(n-3,k-1) - & \\ T(n-4,k-1) & \text{if } n > 3k+1 \end{cases}$$

2-overlapping templates: 0101, 1010.

$$T(n, 0) = 2T(n-1, 0) - T(n-2, 0) + 2T(n-3, 0) - T(n-4, 0)$$

$$T(n,k) = \begin{cases} 0 & \text{if } n < 2k+2 \\ 1 & \text{if } n = 2k+2 \\ 2T(n-1,k) - T(n-2,k) + & \\ 2T(n-3,k) - T(n-4,k) + & \\ T(n-2,k-1) - 2T(n-3,k-1) + & \\ T(n-4,k-1) & \text{if } n > 2k+2 \end{cases}$$

3-overlapping templates: : 0000, 1111.

$$T(n,0) = T(n-1,0) + T(n-2,0) + T(n-3,0) + T(n-4,0)$$

$$T(n,k) = \begin{cases} 0 & \text{if } n < k+3 \\ 1 & \text{if } n = k+3 \\ T(n-1,k) + T(n-2,k) + & \\ T(n-3,k) + T(n-4,k) + & \\ T(n-1,k-1) - T(n-2,k-1) - & \\ T(n-3,k-1) - T(n-3,k-1) & \text{if } n > k+3 \end{cases}$$

Probabilities for each i-overlapping template can be evaluated by

$$F_n(k) = \frac{T(n,k)}{2^n}$$

To construct overlapping template tests, from each i-overlapping template set a representative template is chosen instead of testing all possible templates, and test is done with four selected templates.

5.3 Application

5.3.1 Experimental Parameters and Boundary Values

To describe an application of the health test suite two parameter sets are chosen as follows.

Parameter Set 1: For the window size $W = 1024$ and repetition number $k = 976$, nearly 1 million bits of the sequence can be tested. If significance level of the test suite is determined as $\alpha = 2^{-33}$, for each random variable critical values are evaluated as:

$$\alpha = 1 - (1 - \alpha_i)^7$$

Then for each i, $\alpha_i = 2^{-35}$. Since each random variable is employed 976-times, for each repeated test significance levels β can be evaluated by the following formula:

$$\alpha_i = 1 - (1 - \beta)^{976}$$

Then $\beta \approx 2^{-45}$. By using probability distribution functions or recursions of them for each random variable, valid intervals are chosen as follows. (Since probability distribution functions of random variables are discrete, the boundary values were selected approximately to be closest to $\beta/2$.)

Table 5.1: Health Tests Boundary Values for 1 million bit

Random Variable	**Left Boundary Value**	**Right Boundary Value**	**Significance Level** ($\approx$)
Weight	392	634	2^{-45}
Run	391	633	2^{-45}
Runs of Length 1	133	402	2^{-45}
1-Overlapping Template	22	111	2^{-45}
2-Overlapping Template	17	124	2^{-45}
3-Overlapping Template	14	136	2^{-45}
4-Overlapping Template	4	186	2^{-45}

Parameter Set 2: For the window size $W = 1024$ and repetition number $k = 20$, health tests are employed for nearly 20 000 bit of the sequence. Similarly, significance level of the test suite is determined as $\alpha = 2^{-33}$, for each random variable critical values are evaluated as:

$$\alpha = 1 - (1 - \alpha_i)^7$$

Then for each i, $\alpha_i = 2^{-35}$. Since each random variable is employed 20-times, for each repeated test significance levels β can be evaluated by the following formula:

$$\alpha_i = 1 - (1 - \beta)^{20}$$

Significance level is approximately evaluated as $\beta \approx 2^{-40}$ for each repetition. By using probability distribution functions or recursions of them for each random variable, valid intervals are chosen as follows. (Since probability distribution functions of random variables are discrete, the boundary values were selected approximately to be closest to $\beta/2$.)

Table 5.2: Health Tests Boundary Values for 20 000 bit

Random Variable	**Left Boundary Value**	**Right Boundary Value**	**Significance Level** ($\approx$)
Weight	400	625	2^{-40}
Run	399	624	2^{-40}
Runs of Length 1	140	391	2^{-40}
1-Overlapping Template	24	107	2^{-40}
2-Overlapping Template	19	119	2^{-40}
3-Overlapping Template	16	130	2^{-40}
4-Overlapping Template	5	175	2^{-40}

5.3.2 Experimental Results

The following datasets were simulated for the experiments. Each generated sequence is of length 1 000 000. When health tests are applied with parameters of *Parameter Set 2*, the first 20 000 bit of the sequences are tested, and the remaining bits are omitted.

1. **Uniform distribution without any known bias. AES-128** The sequences are generated using the Cipher Block Chaining (CBC) mode of the block cipher Advanced Encryption Standard (AES) [9]. This dataset contains 200 sequences of length 1 000 000. In these sequences, all outputs are assumed to have an equal probability of occurring, and are independent. Hence, they are assumed to be random.

2. **Uniform distribution without any known bias. QUANTIS** IDQ-QUANTIS [12] is a true random number generator, generation mechanism of QUANTIS is based on quantum physics. To generate sequences, there is no need for seed or input sequence. By using IDQ-QUANTIS 200 binary sequences of length

$1\,000\,000$ are generated. The device is certified to the highest levels of entropy and randomness testing, outputs are assumed to be random.

3. **Biased binary distribution with Pr(1)=0.7 and Pr(0)=0.3.** The dataset follows a biased binary distribution, where the probability of observing a 0 is 0.7, and the probability of observing a 1 is 0.3. To generate this dataset, 200 sequences of length $1\,000\,000$ were generated. This data is generated using the random number generator Mersenne Twister (MT19937) in `C++`.

4. **Biased binary t-bit Duplication** t-bit duplication is defined as copying consecutive t-bit non-overlapping blocks of the sequence to the end of each block. This method doubles the length of the sequence. Initial sequences of length $500\,000$ are generated by CBC-mode of AES-128. Initial sequences are transformed with 128-bit, 256-bit, and 100-bit duplication. Three datasets are generated, each containing 200 binary sequences of length $1\,000\,000$.

Table 5.3: Health tests results for simulated data sets with Parameter Set 1

		Weight	**Run**	**Run1**	**Temp1**	**Temp2**	**Temp3**	**Temp4**
AES CBC	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
QUANTIS	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
Entropy=0.5	**pass**	0	0	200	0	0	200	0
	fail	200	200	0	200	200	0	200
128-dup	**pass**	199	200	200	200	200	200	200
	fail	1	0	0	0	0	0	0
256-dup	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
100-dup	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0

Table 5.4: Health tests results for simulated data sets with Parameter Set 2

		Weight	**Run**	**Run1**	**Temp1**	**Temp2**	**Temp3**	**Temp4**
AES CBC	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
QUANTIS	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
Entropy=0.5	**pass**	0	101	200	0	0	200	5
	fail	200	99	0	200	200	0	195
128-dup	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
256-dup	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0
100-dup	**pass**	200	200	200	200	200	200	200
	fail	0	0	0	0	0	0	0

Health tests are employed for each dataset and numbers of passes and fails of each individual test are given in Table 5.3 and Table 5.4, for different parameters. Experimental results show that nearly all sequences of uniformly distributed datasets generated by AES-128 and IDQ-QUANTIS pass health tests as expected. The sequences of the dataset follow a biased binary distribution, with the probability of observing a 1 being 0.7 and the probability of observing a 0 being 0.3, mostly failing from *weight, run* and *template* tests. *Weight* and *run* tests detect the sequences of the dataset generated by t-bit duplication, with parameter set 1.

In this study, health tests for are examined, these are used to detect corruption and error in the working mechanism of the entropy source of a RNG and give warnings about disruptions in the process. Existing methods are revisited, and a basic model of the health test suite is introduced with random variables *weight, run, runs of length 1* and *overlapping templates*.

CHAPTER 6

CONCLUSION

In cryptography, random numbers are used in almost all applications, and the security of these applications depends on the assumption that these numbers are generated uniformly at random and are unpredictable. In this thesis, we focus on testing random number generators.

Standards and recommendations for randomness testing have been examined, and we focus on statistical methods for entropy estimation of NIST SP 800-90B [32]. The NIST Special Publication (SP) 800-90B, Recommendation for the Entropy Sources Used for Random Bit Generation is analyzed with statistical methods to evaluate accuracy, effectiveness, and limitations of the NIST SP 800-90. In this part of the thesis, accuracy is evaluated by investigating the estimation results for simulated random numbers with known entropy. We analyze the correlation between entropy estimates by using Pearson and Spearman's metrics, we study the impacts of deterministic transformations on the estimators and impact of IID-Assumption tests on estimators.

We expect the provided results to help improve the accuracy of NIST's entropy estimation strategy and promote similar studies to consider the impacts of commonly used conditioning or post-processing functions.

We propose a new black-box, statistical-based entropy estimator called *index-value coincidence estimate*. This estimator is used to estimate both min-entropy and *Shannon entropy* as the technique first estimates the probability distribution. Experiments show that index-value coincidence estimate gives expected and accurate entropy estimations for simulated datasets and may be employed with the other estimators of NIST SP 800-90B to increase the number of estimators and accuracy of results.

Moreover, we investigate the details of existing health tests for TRNGs. We introduce a statistical model by using distributions of random variables *weight, run, runs of length 1* and *overlapping templates* we construct a health test suite to detect corruption and error in the working mechanism of the entropy source and give warnings about disruptions in the process simultaneously, for cryptographic random number generators.

REFERENCES

[1] AIS 20: Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren (Version 3), Report, Bundesamt für Sicherheit in der Informationstechnik (BSI), May 2013, https://www.bsi.bund.de/dok/6618284 (Accessed: 2022-01-01).

[2] AIS 31: Funktionalitätsklassen und Evaluationsmethodologie für physikalische Zufallszahlengeneratoren (Version 3), Report, Bundesamt für Sicherheit in der Informationstechnik (BSI), May 2013, https://www.bsi.bund.de/dok/6618252 (Accessed: 2022-01-01).

[3] E. B. Barker and J. M. Kelsey, SP 800-90A Recommendation for Random Number Generation Using Deterministic Random Bit Generators, Technical report, National Institute of Standards and Technology, June 2015, https://doi.org/10.6028/NIST.SP.800-90Ar1(Accessed: 2022-01-01).

[4] E. B. Barker, J. M. Kelsey, K. A. McKay, A. Roginsky, and M. Sönmez Turan, SP 800 90C Recommendation for Random Bit Generator (RBG) Constructions (3rd Draft), Technical report, National Institute of Standards and Technology, September 2022, https://doi.org/10.6028/NIST.SP.800-90C.3pd(Accessed: 2022-01-01).

[5] Y. Benjamini and Y. Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, Journal of the Royal statistical society: series B (Methodological), 57(1), pp. 289–300, 1995.

[6] D. J. Bernstein, Y.-A. Chang, C.-M. Cheng, L.-P. Chou, N. Heninger, T. Lange, and N. van Someren, Factoring rsa keys from certified smart cards: Coppersmith in the wild, in K. Sako and P. Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013*, pp. 341–360, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, ISBN 978-3-642-42045-0.

[7] R. G. Brown, Dieharder: A random number test suite, Available at: https://webhome.phy.duke.edu/ rgb/General/dieharder.php, 2013, (Accessed: 2022-09-15).

[8] A. Doğanaksoy, F. Sulak, M. Uğuz, O. Şeker, and Z. Akcengiz, Mutual correlation of nist statistical randomness tests and comparison of their sensitivities on transformed sequences, Turkish Journal of Electrical Engineering and Computer Sciences, 25(2), 2017.

[9] M. Dworkin, N. Mouha, and M. S. Turan, Advanced Encryption Standard (AES), Federal Inf. Process. Stds. (NIST FIPS) 197, National Institute of Standards and Technology, Gaithersburg, MD, 2001 (updated 2023).

[10] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman, Mining your ps and qs: detection of widespread weak keys in network devices, in *Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, p. 35, USENIX Association, USA, 2012.

[11] J. E. Hill, SP 800-90B Refinements: Validation Process, Estimator Confidence Intervals, and Assessment Stability, ICMC, 2020.

[12] IDQuantique, Idq quantis, true random number generator based on quantum physics, https:www.idquantique.com/random-number-generation/products/quantis-random-number-generator/ (Accessed: 2024-01-01).

[13] ISO Central Secretary, ISO/IEC 18031:2011 Information technology — Security techniques — Random bit generation, Standard ISO/IEC 18031:2011, International Organization for Standardization, Geneva, CH, 2011, https://www.iso.org/standard/54945.html (Accessed: 2024-01-01).

[14] ISO Central Secretary, ISO/IEC 19790:2012 Information technology — Security techniques — Security requirements for cryptographic modules, Standard ISO/IEC 19790:2012, International Organization for Standardization, Geneva, CH, 2012, https://www.iso.org/standard/52906.html (Accessed: 2024-01-01).

[15] ISO Central Secretary, ISO/IEC 15408-1:2009 Information technology — Security techniques — Evaluation criteria for IT security — Part 1: Introduction and general model, Standard ISO/IEC 15408-1:2009, International Organization for Standardization, Geneva, CH, 2015, https://www.iso.org/standard/50341.html (Accessed: 2024-01-01).

[16] ISO Central Secretary, Information technology — Security techniques — Test and analysis methods for random bit generators within ISO/IEC 19790 and ISO/IEC 15408, Standard ISO/IEC 20543:2019, International Organization for Standardization, Geneva, CH, 2019, https://www.iso.org/standard/68296.html (Accessed: 2024-01-01).

[17] Y. Kim, C. Guyot, and Y.-S. Kim, On the efficient estimation of min-entropy, IEEE Transactions on Information Forensics and Security, 16, pp. 3013–3025, 2021.

[18] O. Koçak, *A unified evaluation of statistical randomness tests and experimental analysis of their relations*, Phd thesis, Middle East Technical University, Ankara, Turkey, 2016, available at `https://etd.lib.metu.edu.tr/upload/12620341/index.pdf`.

[19] A. Lee, M. Smid, and S. Snouffer, Security requirements for cryptographic modules [includes change notices as of 12/3/2002], 2001-05-25 2001, https://doi.org/10.6028/NIST.FIPS.140-2 (Accessed: 2022-01-01).

[20] H. Li, J. Zhang, Z. Li, J. Liu, and Y. Wang, Improvement of min-entropy evaluation based on pruning and quantized deep neural network, IEEE Transactions on Information Forensics and Security, 18, pp. 1410–1420, 2023.

[21] P. L'Ecuyer and R. Simard, Testu01: A c library for empirical testing of random number generators, Available at: http://www.iro.umontreal.ca/ lecuyer/testu01/, 2007, (Accessed: 2022-09-15).

[22] Y. Ma, T. Chen, J. Lin, J. Yang, and J. Jing, Entropy estimation for adc sampling-based true random number generators, IEEE Transactions on Information Forensics and Security, 14(11), pp. 2887–2900, 2019.

[23] G. Marsaglia, The marsaglia random number cdrom including the diehard battery of tests of randomness, Available at: https://www.stat.fsu.edu/ geo/diehard/, 1996, (Accessed: 2022-09-15).

[24] K. Pearson and G. L. for National Eugenics, *"Note on Regression and Inheritance in the Case of Two Parents".*, Proceedings of the Royal Society, Royal Society, 1895.

[25] M. Peter and W. Schindler, A Proposal for Functionality Classes for Random Number Generators (Version 2.35, DRAFT) , Report, Bundesamt für Sicherheit in der Informationstechnik (BSI), September 2022, https://www.bsi.bund.de/dok/ais-20-31-appx-2022 (Accessed: 2023-02-01).

[26] A. Rényi, On measures of entropy and information, in *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability, volume 1: contributions to the theory of statistics*, volume 4, pp. 547–562, University of California Press, 1961.

[27] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, M. L. S. Leigh, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, A statistical test suite for random and pseudo random number generators for cryptographic applications, 2001.

[28] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, N. Heckert, J. Dray, S. Vo, and L. Bassham, SP 800-22 Rev. 1a A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, Technical report, National Institute of Standards and Technology, 2010, https://doi.org/10.6028/NIST.SP.800-22r1a (Accessed: 2022-01-01).

[29] C. E. Shannon, A mathematical theory of communication, The Bell System Technical Journal, 27, pp. 379–423, 1948.

[30] C. Spearman, The proof and measurement of association between two things, American Journal of Psychology, 15, pp. 88–103, 1904.

[31] F. Sulak, M. Uğuz, O. Koçak, and A. Doğanaksoy, On the independence of statistical randomness tests included in the nist test suite, Turkish Journal of Electrical Engineering and Computer Sciences, 25(5), pp. 2205–2217, 2017.

[32] M. Sönmez Turan, E. B. Barker, J. M. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle, SP 800-90B Recommendation for the Entropy Sources Used for Random Bit Generation, Technical report, National Institute of Standards and Technology, January 2018, https://doi.org/10.6028/NIST.SP.800-90B(Accessed: 2020-01-01).

[33] M. S. Turan, A. Doğanaksoy, and S. Boztaş, On independence and sensitivity of statistical randomness tests, In International Conference on Sequences and Their Applications (SETA), pp. 204–215, 2008.

[34] J. Woo, C. Yoo, Y.-S. Kim, Y. Cassuto, and Y. Kim, Generalized lrs estimator for min-entropy estimation, IEEE Transactions on Information Forensics and Security, 18, pp. 3305–3317, 2023.

[35] S. Zhu, Y. Ma, T. Chen, J. Lin, and JiwuJing, Analysis and improvement of entropy estimatorsin nist sp 800-90b for non-iid entropysources, IACR Transactions on Symmetric Cryptology, 2017(3), pp. 151–168, 2017.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Aslan, Melis

EDUCATION

Degree	**Institution**	**Year of Graduation**
Ph.D.	IAM, METU	2024
Minor	Business Administration, METU	2015
B.S.	Mathematics, METU	2015
High School	Sokullu Mehmet Paşa Lisesi	2007

PROFESSIONAL EXPERIENCE

Year	**Place**	**Enrollment**
Jan 2018-	Department of Mathematics, METU	Research Assistant
July 2017-	Fame Crypt	Researcher

PUBLICATIONS

Journal Publications

1. **Q2** "LS-14 test suite for long sequences", Z. Akcengiz, M. Aslan, A. Doğanaksoy, F. Sulak, M.Uğuz, Hacettepe Journal of Mathematics and Statistics, 2024. DOI 10.15672/hujms.1190807

International Conference Publications

1. “Observations on NIST SP 800-90B Entropy Estimators”, M. Aslan, A. Doğanaksoy, Z. Saygı, M.Sönmez Turan, F. Sulak, Sequences and Their Applications, 2024.

2. “Statistical Randomness Test of Long Sequences by Dynamic Partitioning”, Z. Akcengiz, M. Aslan, Ö. Karabayır, A. Doğanaksoy, M. Uğuz, F. Sulak, International Conference on Information Security and Cryptology, 2020.

National Conference Publications

1. “Kriptografik Rastgele Sayı Üreteçleri için Sağlık Testleri”, M. Aslan, A. Doğanaksoy, Z. Saygı, F.Sulak, 15. Ankara Mathematics Days, 2024.

PROJECTS

1. **ASELSAN** Design of Entropy and Health Tests Suite for Random Number Generators Project (2023 - 2024)

2. **TÜBİTAK TEYDEB 1501-** Hardware Security Module Based Transparent Data Encryption and Improved SSL/TLS Design Project (2021 - 2022)

3. **ODTÜ BAP-** Specified Cyrptographic Randomness Test Suite for Block Ciphers and Hash Functions Project (2021 - 2022)

4. **ODTÜ TEKNOKENT-** Cryptanalysis of A5/1 Stream Cipher Project (2021 - 2022)

5. **TÜBİTAK BİLGEM-** Cryptanalysis Consultancy Project (2017 - ...)