# Jump and Hop Randomness Tests for Binary Sequences

Haitao Li, Yang Liu, Ming Su*, and Gang Wang
Department of Computer Science
Nankai University, Tianjin, P. R. China

**Abstract**

Linear complexity test included in the NIST test suite only checks whether or not the observed linear complexity is close to the expected linear complexity. To take full advantage of the information of linear complexity profile, we propose two randomness tests including a jump test based on the jump complexity, i. e., the number of changes in the linear complexity profile of a sequence, and a hop test checking the sum of jump heights at intervals. By using an iterative algorithm we calculate some necessary statistical measures of a random sequence of length $M$, and combining with hypothesis test we determine whether a given binary sequence is random or not. The computational complexity of the jump test and the hop test is the same as that of the linear complexity test. Additionally, we provide a type of sample which, having passed all tests in the NIST test suite, is rejected by the jump test and hop test. So the proposed tests deserve to be considered.

## 1 Introduction

Random sequences are important in secure cryptographic systems, and are widely used in encryption and wireless communications, such as random number generator used in stream ciphers, and authentication in the preliminary stage in the secure handshake protocols. However, if pseudorandom sequences used in such systems show an evidence for nonrandomness, it might give adversaries useful tips to attack these systems.

There are two typical methods to determine the randomness of a given sequence. One is prior test which is more experimental, such as the National Institute of Standards and Technology (NIST) test suite NIST SP800-22 [1], which are useful as a first step in determining whether or not a generator is suitable for a particular cryptographic application. The NIST test suite is based on the method of statistical hypothesis testing and includes 15 types of tests, namely: frequency, block frequency, runs, longest run, matrix rank, DFT (spectral test), non-overlapping template, overlapping template, a "Universal Statistical" Test, linear complexity, serial, approximate entropy, cumulative sums, and two random excursions tests. It was reported that the DFT (spectral test) and the Lempel-Ziv compression test included in the NIST test suite need to be corrected. Then, the linear complexity test is the only one test that can check the difficulty of prediction on sequences. The *linear complexity* is the length of the shortest linear feedback

*Corresponding Author: nksuker@gmail.com

register (LFSR) that can generate the given sequence. The linear complexity profile is also an important cryptographic characteristic of sequences. Compared with the linear complexity, the *linear complexity profile* of a sequence is a measure describing the length of the shortest LFSR that generating the current sequence at each step. Regarding the linear complexity and the linear complexity profile, there are many theoretical research results, see Niederreiter [8, 9, 10, 11], Rueppel [13], M. Wang and J. L. Massey [15], M. Wang [14], and the Berlekamp-Massey algorithm for computing the linear complexity [6]. For truly random binary sequence, i. e., each bit is independent and uniformly distributed, the expectation and the variance of the linear complexity profile were provided, as well as the nice asymptotically behavior [12]. Combining the theoretical results on the distribution of the linear complexity and the statistical null hypothesis testing, the linear complexity was involved in the NIST SP 800-22 test suite.

The other is more theoretical in terms of order of magnitude of random measures. C. Mauduit and A. Sárközy, et al., [3], [5], proposed the notion of well distribution measure, and the correlation measure of order $k$ to determine a give sequence is balanced or uniformly distributed. They also provided the bounds of these measures for a truly random sequence. Accordingly, they can judge whether a sequence is random or not by estimating the order of magnitude of those measures, particularly for those sequence constructed by additive or multiplicative characters, such as a family of binary sequence derived from the Legendre symbol. L. Mérai et al. [7] studied the close relationship between the proposed random measures and the NIST test suite. A. Winterhof discussed the linear complexity and related complexity measures[18].

Considering that the linear complexity in the NIST test suite only detects the behavior the linear complexity profile at the end of the sequence, but not the deviation from the expected line '$y = x/2$', K. Hamano et al. [4] proposed a new randomness test based on the linear complexity profile of sequences, in which the total area of the closed pair of congruent triangles along the line '$y = x/2$' was proposed as a measure for random testing. However, they required that the last point in the linear complexity profile must exist on the line '$y = x/2$', their test become complicated with mixed procedures, and a half of test blocks were wasted and not involved in their remaining test procedure. Therefore, firstly we propose a *jump test* based on the *jump complexity*, i. e., the number of changes in the linear complexity profile of a sequence proposed by Niederreiter [11] and Wang [16], see details in subsection 2.2. Then we propose another test called the *hop test*, checking the sum of jump heights at odd jumps. Accordingly, combining with the hypothesis testing and the $\chi^2$ test on the $P$-values, we design jump test and hop test.

The paper is organized as follows. First we introduce basic concepts in Section 2. Then, we give the detailed procedure of the jump test and the hop test in Section 3. Afterwards, we demonstrate the advantages of our tests by providing a strong example in Section 4.

## 2 Background

### 2.1 Hypothesis Test and Testing Strategies

Let $H_0$ denote the hypothesis that a given binary sequence is independently uniformly distributed over $\{0, 1\}$. A statistical test of randomness will first construct a *test statistic* $X$ on the sequence and then determine the probability distribution of this statistic based on the hypothesis $H_0$. It will produce a $P$-value, which is the probability of obtaining test results at least as extreme as the results actually observed. The *significant level* $\alpha$ of a statistic test is the lower bound of the produced $P$-value to determine whether to accept $H_0$ or not. If the $P$-value is smaller than $\alpha$, then $H_0$ will be rejected. Moreover, under the hypothesis that $H_0$ is true, when many sequences are involved in the statistical test, those computed $P$-values will distribute uniformly in $[0, 1]$. Suppose we test a sample including $s$ sequences and obtain $s$ $P$-values accordingly, we will determine the sample is random or not by evaluating $\mathcal{U}$, where $\mathcal{U}$ is defined to check the uniformity of these $P$-values, i. e., the $P$-value of the $\chi^2$ statistic $\chi^2 = (\sum_{i=1}^{10}(f_i - \frac{s}{10})^2)/\frac{s}{10}$, in which $f_i$ is the number of $P$-values that falls in the interval $C_i = [0.1(i-1), 0.1i)$ for $i = 1, 2, \ldots, 10$. A sample is regarded to be non-random when $\mathcal{U} < 0.0001$.

### 2.2 Linear Complexity and Related Concepts

Considering a sequence $\epsilon^n = \epsilon_0\epsilon_1 \ldots \epsilon_{n-1}$ over $\mathbb{F}_q$, the *linear complexity* $L(\epsilon^n)$ is defined as the length of the shortest linear feedback shift register (LFSR) that can generate $\epsilon^n$, i. e., there exists $L$ elements $c_0, c_1, \ldots, c_{L-1} \in \mathbb{F}_q$ satisfying

$$\epsilon_{i+L} = c_{L-1}\epsilon_{i+L-1} + c_{L-2}\epsilon_{i+L-2} + \cdots + c_0\epsilon_i$$

for $i = 0, 1, \ldots, n-L-1$. Also we define the linear complexity of all zero sequence to be 0 for convention.

An efficient algorithm for computing the linear complexity of a sequence $\epsilon^n$ was designed by James L. Massey [6], derived from an iterative algorithm introduced by Berlekamp for decoding the Bose-Chaudhuri-Hocquenghem codes with the computational complexity $O(n^2)$.

The $i$-th linear complexity $L_i(\epsilon^n), 1 \le i \le n$ of $\epsilon^n$ is the linear complexity of the first $i$ terms of $\epsilon^n$, say $L_i(\epsilon^n) = L(\epsilon_0\epsilon_1 \ldots \epsilon_{i-1})$, and we define $L_0(\epsilon^n) = 0$ for convention. Note that the Berlekamp-Massey Algorithm produces every $L_i(\epsilon^n), 1 \le i \le n$ after accomplishing each loop. We list basic property of the $i$-th linear complexity of $\epsilon^n$ as follows.

**Proposition 1.**

$$L_{i+1}(\epsilon^n) = \begin{cases} L_i(\epsilon^n) & \text{if} \quad L_i(\epsilon^n) > i/2, \\ L_i(\epsilon^n) \quad \text{or} \quad i+1-L_i(\epsilon^n) & \text{if} \quad L_i(\epsilon^n) \le i/2. \end{cases} \tag{1}$$

*For the second case, $L_{i+1}$ equals $L_i$ or $i+1-L_i$ with the same probability $\frac{1}{2}$ for a random binary sequence.*

The *linear complexity profile* of $\epsilon^n$ is defined as the sequential values:$L_1(\epsilon^n), L_2(\epsilon^n), .. L_n(\epsilon^n)$. To vividly explain the linear complexity profile, the *graph of linear complexity profile* is constructed by sequentially connecting the following points

$$(0,\ L_0(\epsilon^n)),\ (1,\ L_0(\epsilon^n)),\ (1,\ L_1(\epsilon^n)), \cdots, (n,\ L_{n-1}(\epsilon^n)),\ (n,\ L_n(\epsilon^n)).$$

An example of a graph of linear complexity profile is shown in Fig. 1, where the line $y = \frac{x}{2}$ is also added.

Because of Proposition 1, there are usually many pairs of congruent triangles in the graph of linear complexity profile, called *PCT*s for short. A PCT is also shown in Fig. 1 denoted by $T_m$, where $m$ is the horizontal width of the two triangles.

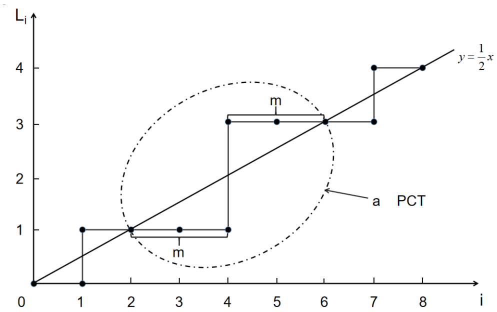



Figure 1: Graph of linear complexity profile and PCT

The jump complexity of a sequence $\epsilon^n$, denoted by $jmp(\epsilon^n)$, is the number of changes ("jumps") in the linear complexity profile of $\epsilon^n$ [17], i. e., $jmp(\epsilon^n)$ is the number of positive integers among $L_1(\epsilon^n), L_2(\epsilon^n) - L_1(\epsilon^n), L_3(\epsilon^n) - L_2(\epsilon^n), \ldots, L_n(\epsilon^n) - L_{n-1}(\epsilon^n)$.

Denote by $J_n$ a random variable for the jump complexity of $\epsilon^n$ over $\mathbb{F}_q$, then Niederreiter [11] calculated the expectation $E(J_n)$ and variance $V(J_n)$ for the variable $J_n$:

$$E(J_n) = \frac{(q-1)n}{2q} + \frac{(q+1)^2 - (-1)^n(q-1)^2}{4(q^2+q)} - \frac{1}{(q+1)q^n}\,, \tag{2}$$

$$V(J_n) = \begin{cases} \frac{(q-1)n}{2q^2} - \frac{q}{(q+1)^2} + \frac{(q-1)n+q}{(q+1)q^{n+1}} - \frac{1}{(q+1)^2q^{2n}} & n \text{ is even}, \\ \frac{(q-1)n}{2q^2} + \frac{q^3-5q^2+q+1}{2q^2(q+1)^2} + \frac{(q^2-1)n+2q^2-q+1}{(q+1)^2q^{n+1}} - \frac{1}{(q+1)^2q^{2n}} & n \text{ is odd}. \end{cases} \tag{3}$$

According to the statistical properties of the linear complexity, the NIST linear complexity test is for determining whether or not a given sequence is complex enough to be considered random, see detailed procedure in [1, Subsection 2.10].

# 3 Proposed New Random Tests

In order to calculate the area of the graph of linear complexity profile, the test proposed by K. Hamano et al. required that the block length $M$ is even and $L_M = \frac{M}{2}$. Accordingly, they had to add extra steps to test the number of blocks satisfied is approximately $\frac{1}{2}$, making the test procedures complicated; and nearly half of a test random sequence is not involved in the $\mathcal{U}$ test. Therefore, we propose the jump test and hop test to take full advantage of the test sequence and the whole linear complexity profile. Carter considered the random tests by using the number of jumps and the frequency of jump heights, but determined the randomness only by one input sequence as a block [2]. In the following, firstly we will provide an iterative algorithm to calculate the probability distribution of the jump complexity, and the sum of jump heights at odd jumps with the block length $M$.

## 3.1 Recursive Calculation for the Exact Distribution

Let $J_M$ be the corresponding random variable of the jump complexity of a binary sequence with length $M$ and $\mathcal{J}_M$ be the probability distribution of $J_M$. Since $J_M$ takes finite discrete values,

$\mathcal{J}_M$ can then be represented by a finite set of pairs as follows:

$$\mathcal{J}_M = \Big\{[d_1, \Pr(J_M = d_1)], [d_2, \Pr(J_M = d_2)], \dots\Big\} \tag{4}$$

In order to calculate $\mathcal{J}_M$, we divide the graph of linear complexity profile of every sequence with length $M$ into the first PCT and the remaining part illustrated in Fig. 2.

Suppose the first PCT to be $T_s$, $1 \le s \le \lfloor \frac{M}{2} \rfloor$, then analyzing $\mathcal{J}_M$ can be reduced to analyzing the probability distribution of the remaining part $\mathcal{J}_{M-2s}$. Also, there is a special case that only an incomplete PCT exists in the whole graph of linear complexity profile, shown in Fig. 3. Analogously, denote by $jmp(T_{M,j})$ the jump complexity for this *incomplete PCT* $T_{M,j}$, where $j$ is the number of the $N$-th linear complexities satisfying $L_N = 0$ for $1 \le N \le M$, then we have $j \in \Big\{\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor + 1, \dots, M\Big\}$.

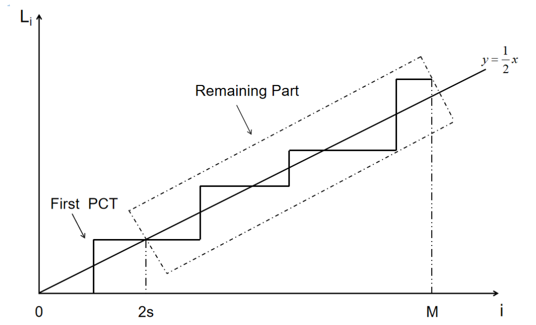



Figure 2: First PCT and remaining part

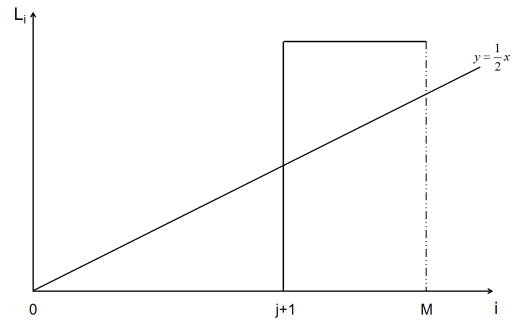



Figure 3: One incomplete PCT $T_{M,j}$ derived from $\epsilon^M$

It is straightforward that

$$jmp(T_{M,j}) = \begin{cases} 1 & j = \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor + 1, \dots, M-1, \\ 0 & j = M. \end{cases} \tag{5}$$

By Proposition 1, for $j = \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor + 1, \dots, M-1$, the probability of the occurrence of $T_{M,j}$ is $(\frac{1}{2})^{j+1}$, and the probability of the occurrence of $T_{M,M}$(all 0s sequence) is $(\frac{1}{2})^M$, which equals that of $T_{M,M-1}$.

In order to calculate $\mathcal{J}_M$ conveniently, we introduce an operator '$\circ$' as follows:

$$[a, b] \circ [c, d] = [a + c, b \times d]$$
$$\Big\{[a_1, b_1], [a_2, b_2], \dots\Big\} \circ [c, d] = \Big\{[a_1, b_1] \circ [c, d], [a_2, b_2] \circ [c, d], \dots\Big\},$$

where the first element of '$[*, *]$' represents a value of a random variable, and the second one represents the occurrence probability of the value.

Then the discrete distribution $\mathcal{J}_M$ can be calculated by

$$\mathcal{J}_M = \Big(\bigcup_{j=1}^{\lfloor \frac{M}{2} \rfloor} \mathcal{J}_{M-2j} \circ [1, \Pr(T_j)]\Big) \bigcup \Big(\bigcup_{j=\lfloor \frac{M}{2} \rfloor}^{M} \Big\{[1, \Pr(T_{M,j})]\Big\}\Big). \tag{6}$$

By (4), (5) and (6), the recursive algorithm for calculating $\mathcal{J}_M$ is as follows:

**Algorithm 1** The algorithm for calculating $\mathcal{J}_M$

**Input:** Sequence length $M$
**Output:** The probability distribution $\mathcal{J}_M$ of the random variable $J_M$

```
𝒥_0 := {[0, 1]},  𝒥_1 := {[0, 1/2], [1, 1/2]}
for all i ∈ {parity(M) + 2, parity(M) + 4, parity(M) + 6, ··· , M} do
    for all j ∈ {1, 2, ··· , ⌊i/2⌋} do
        𝒥_i := 𝒥_i ∪ (𝒥_{i−2j} ∘ [1, (1/2)^j])
    end for
    for all j ∈ {⌊i/2⌋ + 1, ⌊i/2⌋ + 2, . . . , i} do
        𝒥_i := 𝒥_i ∪ {[1, (1/2)^j]}
    end for
    𝒥_i := 𝒥_i ∪ {[0, (1/2)^i]}
end for
Return 𝒥_M.
```

Computing $\mathcal{J}_M$ can be regarded as a bottom-up process and its computational complexity is of polynomial time. Some typical statistical measures of the probability distribution of $J_M$ is shown in Table 1, matching with the explicit expectation and variance in Eq. (2) and Eq. (3).

Table 1: Statistics of the probability distribution of $J_M$

| M | 50 | 100 | 150 | 200 | 500 |
|---|---|---|---|---|---|
| Expectation | 12.8333 | 25.3333 | 37.8333 | 50.3333 | 125.3333 |
| Variance | 6.0278 | 12.2778 | 18.5278 | 24.7778 | 62.2778 |
| Upper 50% | 13 | 25 | 38 | 50 | 125 |
| Upper 5% | 17 | 31 | 45 | 59 | 138 |
| Upper 1% | 18 | 33 | 48 | 62 | 144 |
| Upper 0.1% | 20 | 36 | 51 | 66 | 150 |

Taking the positive integers from the $L_1(\epsilon^M), L_2(\epsilon^M) - L_1(\epsilon^M), \ldots, L_M(\epsilon^M) - L_{M-1}(\epsilon^M)$ and we record them as $\{jh_1(\epsilon^M), jh_2(\epsilon^M), jh_3(\epsilon^M), jh_4(\epsilon^M), \ldots\}$. Define the *odd hop sum* to be $jh_1(\epsilon^M) + jh_3(\epsilon^M) + \cdots$, and the *even hop sum* to be $jh_2(\epsilon^M) + jh_4(\epsilon^M) + \cdots$. It can be seen that the addition of the odd hop sum and the even hop sum of $\epsilon^M$ is exactly $L_M(\epsilon^M)$.

Now let $O_M$ be the random variable of the odd hop sum of $\epsilon^M$, and $E_M$ the even hop sum respectively. The probability distribution of $O_M$ and $E_M$ are denoted by $\mathcal{O}_M$ and $\mathcal{E}_M$ respectively. For calculating $\mathcal{O}_M$ and $\mathcal{E}_M$ recursively, we also divide the linear complexity graph of $\epsilon^M$ into the first PCT and the remaining part. If the first PCT is $T_s$, $1 \le s \le \lfloor \frac{M}{2} \rfloor$, the partial probability distribution $\mathscr{O}_M^{(1)}$ and $\mathcal{E}_M^{(1)}$ can be calculated as follows:

$$\begin{aligned}\mathscr{O}_M^{(1)} &= \bigcup_{s=1}^{\lfloor \frac{M}{2} \rfloor} [s, (\frac{1}{2})^s] \circ \mathcal{E}_{M-2s}, \\ \mathcal{E}_M^{(1)} &= \bigcup_{s=1}^{\lfloor \frac{M}{2} \rfloor} [0, (\frac{1}{2})^s] \circ \mathscr{O}_{M-2s}.\end{aligned} \tag{7}$$

If the whole graph is an incomplete PCT, we have

$$\begin{aligned} ohs(T_{M,j}) &= \begin{cases} j+1 & j = \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor + 1, \ldots, M-1, \\ 0 & j = M, \end{cases} \\ ehs(T_{M,j}) &= 0 \qquad\quad j = \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor + 1, \ldots, M, \end{aligned}$$

where $ohs(T_{M,j})$ is the odd hop sum of the sequence with linear complexity profile graph $T_{M,j}$, and $ehs(T_{M,j})$ the even hop sum.
So in this case, the partial probability distribution $\mathscr{O}_M^{(2)}$ and $\mathcal{E}_M^{(2)}$ can be calculated by

$$\begin{aligned}\mathscr{O}_M^{(2)} &= \Big( \bigcup_{j=\lfloor \frac{M}{2} \rfloor}^{M-1} \big\{ [j+1, (\frac{1}{2})^{j+1}] \big\} \Big) \bigcup \big\{ [0, (\frac{1}{2})^M] \big\}, \\ \mathcal{E}_M^{(2)} &= \bigcup_{j=\lfloor \frac{M}{2} \rfloor}^{M-1} \big\{ [0, (\frac{1}{2})^{j+1}] \big\} \bigcup \big\{ [0, (\frac{1}{2})^M] \big\} = \big\{ [0, (\frac{1}{2})^{\lfloor \frac{M}{2} \rfloor}] \big\}.\end{aligned} \tag{8}$$

Combining (7) and (8) we have

$$\begin{aligned}\mathscr{O}_M &= \mathscr{O}_M^{(1)} \bigcup \mathscr{O}_M^{(2)} = \Big( \bigcup_{s=1}^{\lfloor \frac{M}{2} \rfloor} [s, (\frac{1}{2})^s] \circ \mathcal{E}_{M-2s} \Big) \bigcup \Big( \bigcup_{j=\lfloor \frac{M}{2} \rfloor}^{M-1} \big\{ [j+1, (\frac{1}{2})^{j+1}] \big\} \Big) \bigcup \big\{ [0, (\frac{1}{2})^M] \big\}, \\ \mathcal{E}_M &= \mathcal{E}_M^{(1)} \bigcup \mathcal{E}_M^{(2)} = \Big( \bigcup_{s=1}^{\lfloor \frac{M}{2} \rfloor} [0, (\frac{1}{2})^s] \circ \mathscr{O}_{M-2s} \Big) \bigcup \big\{ [0, (\frac{1}{2})^{\lfloor \frac{M}{2} \rfloor}] \big\}.\end{aligned}$$

Accordingly, we have the following iterative algorithm for computing $\mathscr{O}_M$ and $\mathcal{E}_M$, whose computational complexity is of polynomial time $O(M^4)$.

**Algorithm 2** The algorithm for calculating $\mathscr{O}_M$ and $\mathcal{E}_M$

**Input:** Sequence length $M$

**Output:** The probability distribution of $\mathscr{O}_M$ and $\mathcal{E}_M$

$\mathscr{O}_0 := \left\{[0,1]\right\}, \mathscr{O}_1 := \left\{[0,\frac{1}{2}],[1,\frac{1}{2}]\right\}$

$\mathcal{E}_0 := \left\{[0,1]\right\}, \mathcal{E}_1 := \left\{[0,1]\right\}$

**for all** $i \in \{parity(M)+2, parity(M)+4, parity(M)+6, \cdots, M\}$ **do**

  **for all** $j \in \{1,2,\cdots,\lfloor\frac{i}{2}\rfloor\}$ **do**

    $\mathscr{O}_i := \mathscr{O}_i \bigcup (\mathcal{E}_{i-2j} \circ [j,(\frac{1}{2})^j])$

  **end for**

  **for all** $j \in \{\lfloor\frac{i}{2}\rfloor, \lfloor\frac{i}{2}\rfloor+1,\ldots,i-1\}$ **do**

    $\mathscr{O}_i := \mathscr{O}_i \bigcup \left\{[j+1,(\frac{1}{2})^{j+1}]\right\}$

  **end for**

  $\mathscr{O}_i := \mathscr{O}_i \bigcup \left\{[0,(\frac{1}{2})^i]\right\}$

  **for all** $j \in \{1,2,\cdots,\lfloor\frac{i}{2}\rfloor\}$ **do**

    $\mathcal{E}_i := \mathcal{E}_i \bigcup (\mathscr{O}_{i-2j} \circ [0,(\frac{1}{2})^j])$

  **end for**

  $\mathcal{E}_i := \mathcal{E}_i \bigcup \left\{[0,(\frac{1}{2})^{\lfloor\frac{i}{2}\rfloor}]\right\}$

**end for**

**Return** $\mathscr{O}_M, \mathcal{E}_M$.

Some statistics of the probability distribution of $O_M$ are shown in Table 2.

Table 2: Statistics of the probability distribution of $O_M$

| M | 50 | 100 | 150 | 200 | 500 |
|---|---|---|---|---|---|
| Expectation | 13.1111 | 25.6111 | 38.1111 | 50.6111 | 125.6111 |
| Variance | 6.8395 | 13.0895 | 19.3395 | 25.5895 | 63.0895 |
| Upper 50% | 13 | 26 | 38 | 51 | **126** |
| Upper 5% | 17 | 32 | 45 | 59 | 139 |
| Upper 1% | 19 | 34 | 48 | 62 | 144 |
| Upper 0.1% | 22 | 37 | 52 | 66 | 150 |

**Proposition 2.** *The expectation $E(O_M)$ and variance $V(O_M)$ for the random variable $O_M$ are given by*

$$E(O_M) = \frac{M}{4} + \frac{45-(-1)^M}{72} + \frac{3M-10}{9\cdot 2^M},$$

$$V(O_M) = \begin{cases} \frac{M}{8} + \frac{191}{324} + \frac{27M^2-192M+64}{162\cdot 2^M} - \frac{(3M-10)^2}{81\cdot 4^M}, & M \text{ is even} \\ \frac{M}{8} + \frac{367}{648} + \frac{27M^2-195M+74}{162\cdot 2^M} - \frac{(3M-10)^2}{81\cdot 4^M}, & M \text{ is odd} \end{cases}.$$

We do not present the proof here due to page limitation and will include it in future.

*Remark* 3. Regarding the statistic $O_M$, $E(O_M)$ tends to be close to $\frac{M}{4}$ and $V(O_M)$ tends to be close to $\frac{M}{8}$ when $M$ is large enough. Similarly for the statistic $J_M$, from (2) and (3) we also have

$E(J_M)$ tends to be close to $\frac{M}{4}$ and $V(J_M)$ tends to be close to $\frac{M}{8}$ for binary sequence. However, the jump complexity $J_M$ and the odd hop sum $O_M$ seem to have different characteristics. On the one hand, $O_M$ takes value in a larger range $0 \leq O_M \leq M$ while $0 \leq J_M \leq M/2$. On the other hand, if $J_M$ is small, $O_M$ can be very large, e. g., $J_M = 1$ and $O_M = M$ for the sequence $\underbrace{0\ldots01}_{M}$; and if $J_M$ is large, $O_M$ can be comparatively small, e. g., $J_M = M/2$ and $O_M = M/4$ when considering the sequence with the perfect linear complexity profile $(1,1,2,2,\ldots,M/2,M/2)$ if $M$ is even. Therefore, we consider the following Jump test and Hop test based on different statistics $J_M$ and $O_M$.

### 3.2 Jump Test and Hop Test

The new test is based on the following procedure, where the random variable $X_M$ can be one of the proposed random variables, $J_M$ or $O_M$. We have the jump test when using $J_M$, and the hop test when using $O_M$.

**Procedure of New Test**

**S1** Partition a given binary sequence $\epsilon$ of length $n$ into $N$ disjoint blocks of length $M$, say $\epsilon = \epsilon_1^M \epsilon_2^M \ldots \epsilon_N^M$, where $N = \lfloor n/M \rfloor$ (throwing away the extra bits when $n$ is not the integer times of $M$).

**S2** Let $T$ be the upper 50% point of $X_M$, and $\Phi = \Pr\{X_M > T\}$.

**S3** Calculate $x_i$, which is the observed value of $X_M$ for each $\epsilon_i^M, i = 1,2,\ldots,N$, by using Berlekamp-Massey algorithm.

**S4** Calculate $p = \frac{\#\{x_i | x_i > T\}}{N}$.

**S5** Calculate $z = \frac{p-\Phi}{\sqrt{\frac{\Phi(1-\Phi)}{N}}}$.

**S6** Calculate $P-value = \operatorname{erfc}(\frac{|z|}{\sqrt{2}})$, where $\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt.$

Note that the value of $\Phi$ above is not exactly 50% due to the discreteness of the probability distribution of $X_M$. For example, for the upper 50% point of $J_{500}$, we have $T = 125$ and $\Phi = \Pr\{J_{500} > T\} = 0.491568$.

If $H_0$ is true, the random variable $Z$ should obey the standard normal distribution according to the De Moivre-Laplace theorem in probability theory. Then we have

$$\operatorname{erfc}(\frac{|z|}{\sqrt{2}}) = \frac{2}{\sqrt{\pi}} \int_{\frac{|z|}{\sqrt{2}}}^\infty e^{-t^2} dt = \frac{2}{\sqrt{2\pi}} \int_{|z|}^\infty e^{-\frac{u^2}{2}} du.$$

Thus, we have $\operatorname{erfc}(\frac{|z|}{\sqrt{2}}) = 2\Pr\{Z > |z|\}$, implying $\operatorname{erfc}(\frac{|z|}{\sqrt{2}})$ can be used as the $P$-value of the random variable $Z$.

In order to apply the jump test or hop test, we need to calculate the jump complexity or odd hop sum of a sequence (see step S3) in each block, with computational complexity $O(M^2)$ by using the Berlekamp-Massey algorithm. Thus, the computational complexity of our new test is $O(M^2) \cdot \frac{n}{M} = O(M \cdot n)$, almost the same as that of NIST linear complexity test. We also experimentally provide comparison for time cost of these tests in Table 3. The running CPU is Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz, total memory is 251G bytes, and we take $M = 500$. From Table 3 we see that the time costs of these tests are approximately the same, and they are linear dependent with the size $n$.

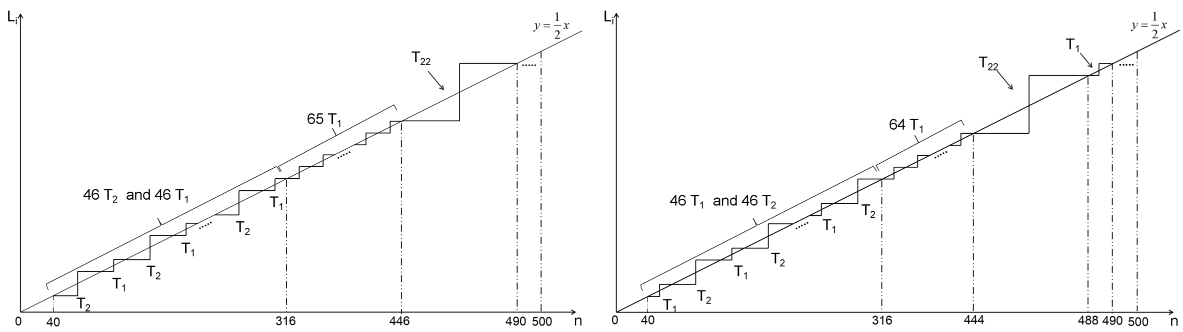
Table 3: Time costs of the mentioned tests (in seconds)

| Size ($n$ bits) / Test | $2^{24}$ | $2^{25}$ | $2^{26}$ | $2^{27}$ | $2^{28}$ | $2^{29}$ | $2^{30}$ |
|---|---|---|---|---|---|---|---|
| NIST LC Test | 1.49 | 3.03 | 5.91 | 10.71 | 21.37 | 43.02 | 83.86 |
| Jump Test | 1.34 | 2.83 | 5.38 | 10.90 | 20.97 | 41.79 | 83.66 |
| Hop Test | 1.47 | 2.76 | 5.34 | 10.93 | 22.61 | 44.33 | 85.54 |

## 4 Experimental Results

In the following tests, the block size $M$ is set to be as suggested, and the length of a sample is $10^9$. A sample will be divided into 1000 sequences of length $10^6$ and will produce 1000 $P$-values. The final decision is derived from these 1000 $P$-values.

We construct a sample that passes all the NIST tests, but is rejected by the Jump test and Hop test, shown in Fig. 4. For every block of length 500, we first fill the whole block with a random sequence. Then we take necessary adjustment on each block, setting its linear complexity profile from 40th bit to 490th bit to be the format in Fig. 4(a) or 4(b), depending on the parity of the number of jumps in the first 40 bits. The last 10 bits of each block are filled with random bits.



(a) The number of jumps in the first 40 bits is Even (b) The number of jumps in the first 40 bits is Odd

Figure 4: Example-Passing NIST test suite but rejected by Jump test and Hop test

Observing the linear complexity profile of each block, the jump complexity of each block is at least $111 + 46 + 1 = 158$, much larger than the selected upper point $T_J = 125$, so the

$P$-value is zero for each sequence. Moreover, the odd hop sum for each sequence is at least $46\times 2+32\times 1+1\times 22=146$, much larger than the selected upper point $T_H=126$. This is the reason why each sequence is rejected by the jump test and the hop test.

Actually, we can construct many samples having the same format of linear complexity profile in Fig. 4(a) or 4(b), by using the continued fractions and the increment sequence derived from the linear complexity profile [14], and the number of them can be estimated at least $2^{111\times 1+46\times 2+22}$, when only considering the middle 450 bits. Accordingly, we have designed a scheme generating 500 random bits in each block first, then adjust each sequence obeying the same linear complexity profile format in Fig. 4(a) or 4(b), and obtain a sample of $10^9$ bits. Very interestingly, the constructed sample passes all tests included in the NIST test Suite (188 tests)! But it is rejected by the jump test and hop test.

*Remark* 4. Randomness tests for pseudorandom generators usually include a random measure with low computational complexity to deal with test sequences of large length. Moreover, the exact distribution of the random measure should be determined in order to combine with statistical hypothesis test. Besides, some nice pseudorandom sequence like Legendre sequence still passes the NIST test suite, as well as the jump test and the hop test.

Let $p$ be a prime. The $p$-periodic Legendre sequence is defined by

$$\ell_i=\begin{cases}\left[1+\left(\frac{i}{p}\right)\right]\Big/2, & \text{if } i\neq 0 \pmod p\\ 0, & \textit{otherwise}\end{cases} \tag{9}$$

We have the following conjecture, revealing that the value of jump complexity $J_M$ and the value of odd hop sum $O_M$ of Legendre sequence do not fluctuate much, this is an explanation why Legendre sequence passes the jump test and the hop test.

**Conjecture 5.** For the block size $M$ large enough (e. g., $M\geq 500$ as NIST suggests), considering the part of Legendre sequence (9) with start position $i\geq 0$ of length $M$ $(\ell_i,\ldots,\ell_{i+M-1})$, we denote by $J_{i,M,p}$ the number of jumps of this partial Legendre sequence, and $O_{i,M,p}$ the odd hop sum of this partial Legendre sequence. Then for any $p>M$, and $i_1,i_2\geq 0$ we have

$$\left|J_{i_1,M,p}-J_{i_2,M,p}\right|<\frac{M}{4},$$
$$\left|O_{i_1,M,p}-O_{i_2,M,p}\right|<\frac{M}{4}.$$

We apply the NIST random test suite to Legendre sequence with $p=2^{31}-1$, and take $n=10^9$, $M=500$. We divide the test sample into 1000 sequences and calculate one P-value for each sequence. The test results is shown in Table 4, including Jump Test and Hop Test. The minimum passing rate for each statistical test is approximately = 980 for a sample of 1000 sequences, and the minimum passing $\mathcal{U}$ value of these 1000 P-values for each statistical test is 0.0001. Thus, the Legendre sequence passes listed partial NIST tests as well as our tests.

Table 4: Random test results for Legendre sequence with $p = 2^{31} - 1$

| C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | P-VALUE | PROPORTION | STATISTICAL TEST |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 94 | 111 | 105 | 114 | 91 | 95 | 105 | 108 | 93 | 84 | 0.457825 | 991/1000 | Frequency |
| 113 | 104 | 91 | 100 | 97 | 101 | 92 | 96 | 106 | 100 | 0.916599 | 987/1000 | BlockFrequency |
| 96 | 97 | 90 | 125 | 101 | 90 | 105 | 88 | 103 | 105 | 0.308561 | 987/1000 | CumulativeSums |
| 103 | 101 | 93 | 116 | 107 | 84 | 107 | 98 | 86 | 105 | 0.442831 | 992/1000 | CumulativeSums |
| 95 | 92 | 105 | 109 | 91 | 101 | 100 | 91 | 83 | 133 | 0.043368 | 987/1000 | Runs |
| 94 | 92 | 97 | 126 | 86 | 116 | 84 | 92 | 116 | 97 | 0.032705 | 989/1000 | LongestRun |
| 101 | 108 | 109 | 89 | 97 | 115 | 111 | 105 | 70 | 95 | 0.072964 | 990/1000 | Rank |
| 100 | 110 | 99 | 104 | 109 | 86 | 104 | 81 | 92 | 115 | 0.304126 | 992/1000 | FFT |
| 105 | 87 | 97 | 89 | 117 | 102 | 89 | 102 | 106 | 106 | 0.520102 | 982/1000 | OverlappingTemplate |
| 105 | 98 | 98 | 97 | 106 | 104 | 83 | 101 | 103 | 105 | 0.899171 | 991/1000 | Universal |
| 108 | 101 | 95 | 89 | 96 | 113 | 96 | 104 | 89 | 109 | 0.709558 | 984/1000 | ApproximateEntropy |
| 93 | 105 | 88 | 99 | 107 | 94 | 107 | 91 | 106 | 110 | 0.769527 | 993/1000 | Serial |
| 86 | 93 | 106 | 100 | 95 | 117 | 108 | 99 | 94 | 102 | 0.637119 | 991/1000 | Serial |
| 83 | 104 | 100 | 80 | 105 | 109 | 96 | 118 | 101 | 104 | 0.231956 | 994/1000 | LinearComplexity |
| 87 | 111 | 100 | 118 | 96 | 104 | 94 | 98 | 91 | 101 | 0.566688 | 991/1000 | **Jump Test** |
| 108 | 92 | 79 | 107 | 84 | 105 | 98 | 99 | 107 | 121 | 0.124476 | 993/1000 | **Hop Test** |

## 5 Acknowledgement

The authors would like to express their sincere thanks to the anonymous reviewers for providing valuable suggestions. They are partially supported by Tianjin Key Research and Development Project 19YFZCSF00900.

## References

[1] A statistical test suite for random and pseudorandom number generators for cryptographic applications, special publication 800-22. Tech. rep.

[2] Carter, G.D.: Aspects of Local Linear Complexity. Ph. D. thesis, University of London (1989)

[3] Christian Mauduit, A.S.: On finite pseudorandom binary sequences i: Measure of pseudo-randomness, the legendre symbol. Acta Arithmetica 82(4), 365–377 (1997)

[4] Hamano, K., Sato, F., Yamamoto, H.: A new randomness test based on linear complexity profile. IEICE Transactions 92-A(1), 166–172 (2009)

[5] Julien Cassaigne, Christian Mauduit, A.S.: On finite pseudorandom binary sequences vii: The measures of pseudorandomness. Acta Arithmetica 103(2), 97–118 (2002)

[6] Massey, J.L.: Shift-register synthesis and BCH decoding. IEEE Trans. Information Theory 15(1), 122–127 (1969)

[7] Mérai, L., Rivat, J., Sárközy, A.: The measures of pseudorandomness and the NIST tests. In: Number-Theoretic Methods in Cryptology - First International Conference, NuTMiC 2017, Warsaw, Poland, September 11-13, 2017, Revised Selected Papers. pp. 197–216 (2017)

[8] Niederreiter, H.: Sequences with almost perfect linear complexity profile. In: Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of of Cryptographic Techniques, Amsterdam, The Netherlands, April 13-15, 1987, Proceedings. pp. 37–51 (1987)

[9] Niederreiter, H.: The probabilistic theory of linear complexity. In: Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings. pp. 191–209 (1988)

[10] Niederreiter, H.: Keysystem sequences with a good linear complexity profile for every strating point. In: Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings. pp. 523–532 (1989)

[11] Niederreiter, H.: The linear complexity profile and the jump complexity of keystream sequences. In: Advances in Cryptology - EUROCRYPT '90, Workshop on the Theory and Application of of Cryptographic Techniques, Aarhus, Denmark, May 21-24, 1990, Proceedings. pp. 174–188 (1990)

[12] Rueppel, R.A.: Linear complexity and random sequences. In: Advances in Cryptology - EUROCRYPT '85, Workshop on the Theory and Application of of Cryptographic Techniques, Linz, Austria, April 1985, Proceedings. pp. 167–188 (1985)

[13] Rueppel, R.A.: Analysis and Design of Stream Ciphers. Berlin, Germany: Springer-Verlag (1986)

[14] Wang, M.: Linear complexity profiles and continued fractions. In: Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings. pp. 571–585 (1989)

[15] Wang, M., Massey, J.L.: The characterization of all binary sequences with perfect linear complexity profiles. In: EUROCRYPT. pp. 35–36 (1986)

[16] Wang, M.: Cryptographic Aspects of Sequence Complexity Measures, Ph.D. dissertation. ETH Zurich (1988)

[17] Wang, M.: Linear complexity profiles and jump complexity. Information Processing Letters 61(3), 165 – 168 (1997)

[18] Winterhof, A.: Linear complexity and related complexity measures. In: Selected Topics in Information and Coding Theory. pp. 3–40. World Scientific, Singapore (2010)