On NIST's Compression Estimate Test

P. R. Mishra[*] Bhartendu Nandan[†] Navneet Gaba[‡]

Abstract

In this paper we present our observations about NIST's Compression estimate test given in SP-800 90B. We observe that steps 4 and 7 of the test may be re-framed to gain efficiency. Based on our observations, we propose a modified algorithm for the test which is twice as fast as the NIST's algorithm. We further claim that the values of probability and *min-entropy* in the example given for the test are incorrect. We also provide computational evidence in support of this claim.

1 Introduction

NIST's compression estimate test is one of the ten tests given in NIST's special publication SP-800 90B [1] for entropy estimation for non-IID data. This test is based on the Compression estimate proposed by Hagerty and Draper [2]. It calculates the entropy rate of a dataset, based on the extent to which it can be compressed.

This test mainly comprises five steps viz.,

i. partitioning the data into two disjoint groups

ii. creating a dictionary with help of the first partition

iii. test the other partition with respect to the created dictionary

[*]prasanna.r.mishra@gmail.com
[†]bhartendun@gmail.com
[‡]navneetgaba2000@gmail.com

iv. computing a modified value of mean with help of mean and standard deviation

v. solving an equation through binary search in order to find the probability

We propose an alternate formulation of step iii. and step v. Using this formulation we present a modified algorithm for Compression Estimate Test. We show that our algorithm is twice as fast as the NIST's algorithm. We also indicate an error in the example given after description of the test and provide computational evidence in support of our findings.

The paper is structured in the following manner.

In the next section we provide description of Compression Estimate Test given in NIST's special publication SP-800 90B [1]. In the third section, we provide an alternate formulation of step iii. and step v. as described above. In the fourth section we show that the values of probability (p) and the *min_entropy* given in the example of the test are incorrect and provide the correct values upto 4 places of the decimal. The next i.e., the fifth section contains an algorithm for the test based on our findings and the computation of its complexity. We take NIST's implementation of the test as a benchmark and compare the timings for different data sets.

2 NIST's Description of Compression Estimate Test[1]

Given the input $S = (s_1, \ldots, s_L)$, where $s_i \in A = \{0, 1\}$,

1. Let $b = 6$. Create a new sequence, $S' = (s'_1, \ldots, s'_{\lfloor L/b \rfloor})$, by dividing S into non-overlapping b-bit blocks. If L is not a multiple of b, discard the extra data.

2. Partition the dataset, S', into two disjoint groups. These two groups will form the dictionary and the test data.

   a. Create the dictionary from the first $d = 1000$ elements of S', $(s'_1, \ldots, s'_d)$.

b. Use the remaining $v = \lfloor L/b \rfloor - d$ observations, $(s'_{d+1}, \ldots, s'_{\lfloor L/b \rfloor})$, for testing.

3. Initialize the dictionary *dict* to an all zero array of size 2^b. For i from 1 to d, let $dict[s'_i] = i$. The value of $dict[s'_i]$ is the index of the last occurrence of each s'_i in the dictionary.

4. Run the test data against the dictionary created in Step 2.

   a. Let D be a list of length v.

   b. For i from $d + 1$ to $\lfloor L/b \rfloor$:

      i. If $dict[s'_i]$ is non-zero, then $D_{i-d} = i - dict[s'_i]$. Update the dictionary with the index of the most recent observation, $dict[s'_i] = i$.

      ii. If $dict[s'_i]$ is zero, add that value to the dictionary, i.e., $dict[s'_i] = i$. Let $D_{i-d} = i$.

5. Calculate the sample mean $\bar{X}$, and the sample standard deviation $(\hat{\sigma})$, of $(\log_2(D_1), \ldots, \log_2(D_v))$.

$$\bar{X} = \frac{\sum_{i=1}^{v} \log_2 D_i}{v},$$
$$c = 0.5907$$

and

$$\hat{\sigma} = c\sqrt{\frac{\sum_{i=1}^{v} (\log_2 D_i)^2}{v-1} - \bar{X}^2}.$$

6. Compute the lower-bound of the confidence interval for the mean, based on a normal distribution [3] with a confidence level of 99 %,

$$\bar{X}' = \bar{X} - 2.576\frac{\hat{\sigma}}{\sqrt{v}}$$

7. Using a binary search (bisection method [4]), solve for the parameter p, such that the following equation is true:

$$\bar{X}' = G(p) + (2^b - 1)G(q) \tag{1}$$

where

$$G(z) = \frac{1}{v} \sum_{t=d+1}^{L} \sum_{u=1}^{t} (\log_2 u) F(z, t, u),$$

$$F(z,t,u) = \begin{cases} z^2(1-z)^{u-1} \text{ if } u < t \\ z(1-z)^{t-1} \text{ if } u = t, \end{cases}$$

and

$$q = \frac{1-p}{2^b - 1}. \tag{2}$$

The bounds of the binary search should be 2^{-b} and 1.

8. If the binary search yields a solution, then the *min-entropy* estimation is the negative logarithm of the parameter, p:

$$\textit{min-entropy} = -\log_2(p)/b.$$

If the search does not yield a solution, then the *min-entropy* estimation is:

$$\textit{min-entropy} = \log_2(2) = 1.$$

3 Our Observations and Reformulations

We have observed that there are logical errors and redundancies in the description for Compression Test proposed by NIST. Moreover, there are scopes of improvement at various places to gain efficiency. A significant error in the example has also been observed. These are described in subsequent subsections.

3.1 Testing the Data against the Dictionary Created

Refer to 4.b. of section 2. As per the description, if for some integer $i \in [d+1, \lfloor L/b \rfloor]$, $dict[s'_i]$ is non-zero, the values of arrays D and $dict$ are updated as $D_{i-d} = i - dict[s'_i]$ and $dict[s'_i] = i$. In the other case, when $dict[s'_i] = 0$, the array D is updated as $D_{i-d} = i$ and the value i is added to the dictionary i.e. $dict[s'_i] = i$.

The dictionary is an array of size 2^b initialised with zeros. Note that the size of dictionary is decided at the beginning of the test and it remains constant throughout. It means adding a new value to the dictionary array is equivalent to updating it. With this observation it becomes clear that the condition i. and ii. of 4.b. can be clubbed in a single condition without using *if*.

The restatement of 4.b. is as under:
For i from $d+1$ to $\lfloor L/b \rfloor$: Set $D_{i-d} = i - dict[s'_i]$ and $dict[s'_i] = i$.

3.2 Calculation of $\bar{X}$ and $\hat{\sigma}$

Using the restatement of 4.b as given in section 3.1, it can be shown that creation of list D of size v may be avoided for calculation of $\bar{X}$ and $\hat{\sigma}$. For the computation of these quantities we require two summations viz., $\sum_{i=1}^{v} \log_2 D_i$ and $\sum_{i=1}^{v} (\log_2 D_i)^2$. We have,

$$\sum_{i=1}^{v} \log_2 D_i = \sum_{i=d+1}^{\lfloor L/b \rfloor} \log_2 D_{i-d} = \sum_{i=d+1}^{\lfloor L/b \rfloor} \log_2 (i - dict[s'_i]).$$

Similarly,

$$\sum_{i=1}^{v} (\log_2 D_i)^2 = \sum_{i=d+1}^{\lfloor L/b \rfloor} (\log_2 (i - dict[s'_i]))^2.$$

From the above expression it is clear that the two terms can be computed with the loop in step 4.b. of section 3.1, and the creation of list D is not necessary. Further, for simplicity, the constants $c = 0.5907$ and 2.576 used in steps 5 and 6 of section 2 can be combined to get a single constant, which is 1.5216.

3.3 Modification in Expression for function G

The function G defined in section 2 contains a double summation over indices t and u. We observe that this double summation can be written as a single summation with lesser number of summands. We have

$$G(z) = \frac{1}{v} \sum_{t=d+1}^{L} \sum_{u=1}^{t} (\log_2 u) F(z, t, u) \tag{3}$$

where

$$F(z,t,u) = \begin{cases} z^2(1-z)^{u-1} \text{ if } u < t \\ z(1-z)^{t-1} \text{ if } u = t \end{cases} \tag{4}$$

We define function $T(z,u)$ as,

$$T(z,u) = z(1-z)^{u-1} \tag{5}$$

Using (5),(4) can be written as

$$F(z,t,u) = \begin{cases} zT(z,u) \text{ if } u < t \\ T(z,u) \text{ if } u = t \end{cases} \tag{6}$$

We have from (3),

$$\begin{aligned} G(z) &= \frac{1}{v}\sum_{t=d+1}^{L}\left(\sum_{u=1}^{t-1}(\log_2 u)F(z,t,u) + (\log_2 t)F(z,t,t)\right) \\ &= \frac{1}{v}\sum_{t=d+1}^{L}\sum_{u=1}^{t-1}(\log_2 u)zT(z,u) + \frac{1}{v}\sum_{t=d+1}^{L}(\log_2 t)T(z,t)) \end{aligned} \tag{7}$$

Consider the summation $\sum_{t=2}^{L}\sum_{u=1}^{t-1}(\log_2 u)zT(z,u)$. Observe that in the summation, given a value of $1 \leq u < L$, say k, the term $(\log_2 k)zT(z,k)$ will occur once for each $t > k$. Therefore, for each k s.t. $1 \leq k < L$, this term will be repeated $L-k$ times in the summation. Consequently,

$$\sum_{t=2}^{L}\sum_{u=1}^{t-1}(\log_2 u)zT(z,u) = \sum_{k=1}^{L-1}(L-k)(\log_2 k)zT(z,k) \tag{8}$$

From (8),

$$\begin{aligned} &\sum_{t=d+1}^{L}\sum_{u=1}^{t-1}(\log_2 u)zT(z,u) \\ &= \sum_{k=1}^{L-1}(L-k)(\log_2 k)zT(z,k) - \sum_{k=1}^{d-1}(d-k)(\log_2 k)zT(z,k) \\ &= \sum_{k=d+1}^{L}(L-k)(\log_2 k)zT(z,k) + (L-d)\sum_{k=1}^{d}(\log_2 k)zT(z,k) \end{aligned} \tag{9}$$

From (7) and (9) we have

$$\begin{aligned} G(z) &= \frac{1}{v}\sum_{k=d+1}^{L}\left((L-k)(\log_2 k)zT(z,k)+(\log_2 k)T(z,k)\right) \\ &\quad +\frac{L-d}{v}\sum_{k=1}^{d}(\log_2 k)zT(z,k) \end{aligned}$$

Since $\log_2 1 = 0$, we have

$$G(z) = \frac{1}{v}\left[\sum_{k=d+1}^{L}\left((L-k)z+1\right)(\log_2 k)T(z,k) + z(L-d)\sum_{k=2}^{d}(\log_2 k)T(z,k)\right] \tag{10}$$

The double summation in (3) contains $\frac{(L-d)(L+d+1)}{2}$ terms whereas the new expression (10) has $L-1$ terms only. As $d << L$, the computational complexity of G gets reduced from $O(L^2)$ to $O(L)$. As computation of G is the most intensive step of the algorithm, execution of the whole algorithm becomes significantly faster with this reformulation.

3.4 Error in Example of the Test

NIST has given a worked out example of this test with a shorter sequence on page 47 of [1]. In this example a binary sequence of 48 bits is taken. The value of d is taken as 4. We have experimentally verified that the calculated values are correct upto point 6 of section 2. The values of p and min-entropy are incorrect. The value of p depends on four parameters viz., $\bar{X}', b, v$ and d. From (1) and (2), it is clear that p is an approximate value of p satisfying $L(p, \bar{X}', b, v, d) = 0$ where,

$$L(p, \bar{X}', v, d) = \bar{X}' - G(p) - (2^b - 1)G\left(\frac{1-p}{2^b-1}\right).$$

For the example under consideration, $b = 6, v = 4, d = 4, \bar{X}' = 1.4617$.
As per example, the p satisfying $L(p, \bar{X}', b, v, d) = 0$ for the above quoted values of the parameters is 0.5715. A simple calculation shows that

$$L(0.5715, 1.4617, 6, 4, 3) = -22.4135.$$

We have calculated the value of p as 0.9578. For this value of p, we have

$$L(0.9578, 1.4617, 6, 4, 3) = -0.0016.$$

Clearly, the value of p calculated by us is much more accurate. Accordingly, the value of min-entropy comes out to be 0.0059.

4 Our algorithm and experimental results

Based on our observations, we propose the modified algorithm for computation of min-entropy.
Given the input $S = (s_1, \ldots, s_L)$, where $s_i \in A = \{0, 1\}$,

1. Let $b = 6$. Create a new sequence, $S' = (s'_1, \ldots, s'_{\lfloor L/b \rfloor})$, by dividing S into non-overlapping b-bit blocks. If L is not a multiple of b, discard the extra data.
2. Partition the dataset, S', into two disjoint groups. These two groups will form the dictionary and the test data.
   a. Create the dictionary from the first $d = 1000$ elements of S', $(s'_1, \ldots, s'_d)$.
   b. Use the remaining $v = \lfloor L/b \rfloor - d$ observations, $(s'_d, \ldots, s'_{\lfloor L/b \rfloor})$, for testing.
3. Initialize the dictionary *dict* to an all zero array of size 2^b. For i from 1 to d, let $dict[s'_i] = i$. The value of $dict[s'_i]$ is the index of the last occurrence of each s'_i in the dictionary.
4. Run the test data against the dictionary created in Step 2 as:
   a. Initialize *sum* and *sum_of_square* to zero.
   b. For i from $d + 1$ to $\lfloor L/b \rfloor$:
      i. Set $Q = \log_2(i - dict[s'_i])$ and $dict[s'_i] = i$.
      ii. Update *sum* and *sum_of_square* as,
      $sum = sum + Q$

and

$sum\_of\_square = sum\_of\_square + Q * Q.$

5. Calculate the sample mean $\bar{X}$, and the sample standard deviation ($\hat{\sigma}$) as
$$\bar{X} = \frac{sum}{v},$$
and
$$\hat{\sigma} = \sqrt{\frac{sum\_of\_square}{v} - \bar{X}^2}.$$

6. Compute the lower-bound of the confidence interval for the mean, based on a normal distribution with a confidence level of 99 %,
$$\bar{X}' = \bar{X} - 1.5216\frac{\hat{\sigma}}{\sqrt{v}}$$

7. Using a binary search, solve for the parameter p, such that the following equation is true:
$$\bar{X}' = G(p) + (2^b - 1)G(q) \tag{11}$$
where $G(z)$ is:
$$\frac{1}{v}\left[\sum_{k=d+1}^{L}((L-k)z+1)(\log_2 k)z(1-z)^{k-1} + z(L-d)\sum_{k=2}^{d}(\log_2 k)z(1-z)^{k-1}\right]$$
and
$$q = \frac{1-p}{2^b - 1}.$$
The bounds of the binary search should be 2^{-b} and 1.

8. If the binary search yields a solution, then the *min-entropy* is given as:
$$\textit{min-entropy} = -\log_2(p)/b.$$
If the search does not yield a solution, then the *min-entropy* estimation is 1.

Remark. *We have observed a descriptive error in step 8 of section 2. As per step 8, if the binary search yields a solution, then the min-entropy estimation is the negative logarithm of the parameter, p. It means when the binary search yields a solution, then the min-entropy is $-\log_2(p)$, which is incorrect. We have suitably modified this in our algorithm.*

4.1 Comparison of Timings

To compare efficiency of our algorithm, five sets of binary sequences of different lengths were taken. On each of the five sets NIST algorithm and our algorithm were run. The experiments were performed on an i-7 machine with 4GB of RAM. The timings are compared in table 1. Please note that $d = 100$ is used for the purpose of efficiency test only.

S.No.	Length of sequences (in bits)	Number of Sequences	d	Time taken in secs	
				NIST Algorithm	Our Algorithm
1	900000	10	100	186.92	79.02
2	90000	100	100	149.18	61.41
3	900000	10	1000	190.63	81.69
4	90000	100	1000	150.75	61.19
5	60000	10	1000	9.82	4.05

Table 1: Comparison of timings of NIST's algorithm and our algorithm

Remark. *The comparison of timings shows that our implementation (as per modified algorithm) is twice as fast as the NIST's algorithm. The improvement from quadratic to linear complexity (as discussed in section 3.3) is not visible in the table for timings. The reason is that we have used NIST's implementation of Compression Test for comparison in which they have used certain optimisation to avoid quadratic complexity.*

5 Conclusion

Our present and the earlier work [5] on NIST's Non-IID Tests suggest that there are logical and descriptive errors as well as some redundancies even in final version of NIST document "Recommendation for the Entropy Sources Used for Random Bit Generation" (NIST Special Publication 800-90B, January, 2018). Moreover, at various places there are scopes of improvement in order to make the tests more efficient.

We have also shown that a major step in the Compression Test proposed by NIST may be reformulated in order to reduce its time complexity from quadratic to linear. However, in NIST's implementation of the Compression Test, certain optimisation has been used to gain efficiency. Still, the comparison of timings shows that our algorithm is twice as fast as the NIST's algorithm. We hope that this paper will be useful for experts and researchers.

References

[1] Meltem Sonmez Turan, Elaine Barker, John Kelsey, Kerry A. McKay, Mary L. Baish, Mike Boyle; Recommendation for the Entropy Sources Used for Random Bit Generation, NIST Special Publication 800-90B, January, 2018.

[2] P. Hagerty and T. Draper; Entropy Bounds and Statistical Tests, NIST Random Bit Generation Workshop, December 2012, https://csrc.nist.gov/csrc/media/events/random-bit-generation-workshop-2012/documents/hagerty_entropy_paper.pdf.

[3] K. Krishnamoorthy, Handbook of Statistical Distribution with Applications,Chapman and Hall, 2006.

[4] Richard L. Burden, J. Douglas Flairs, Numerical Analysis, 8th Edition, Brooks/Cole Cengage Learning.

[5] P. R. Mishra, Bhartendu Nandan, Navneet Gaba, An Efficient and Compact Reformulation of NIST's Collision Estimate Test, available on IACR eprint archive.