Binary Matrix Rank Test

Nicu Neculache, Vlad Petcu

January 2022

Abstract

In this paper we study one of the NIST 800-22 Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications [1]. We give an overview for the statistical testing and its importance in cryptography, then we focus on one of the NIST tests, specifically the **Binary Matrix Rank Test**. We describe the logical schema and our own code implementation. Then we evaluate the test by running it on some well chosen test vectors and gathering the results, based on which analysis we do an assumption. More exactly, we validate if the binary sequence input can be classified as random or not.

1 Introduction

Statistical testing is a mathematical technique for analysing an algorithm based on some input-output pairs, usually the inputs being called testing samples. The logical flow consists of running the algorithm (or the system) multiple times, on a significant inputs collection, obtaining the results and analysing them in order to classify/validate the algorithm. Statistical tests are efficient on establishing the ownership of a set of independent observations, or measurements, to a specific population or probability distribution.

Mathematically speaking, a statistic test describes how closely the distribution of the data matches the distribution predicted under the null hypothesis of the statistical test used. The distribution of data is how often each observation occurs, and can be described by its central tendency and variation around that central tendency.

The statistical test are commonly used in the field of cryptography, specifically for the encryption, decryption and the keys or sub-keys generation. Usually, these three processes are strongly dependent on the randomness of the used algorithms (or on the strength of sequence generated against cryptography analysis). This aspect brings up the importance for estimating the entropy, which is a measure of the amount of information needed for an attacker to find the encryption key or to predict the nonce values. If one statistical test finds some predictable information in the analyzed sample, then it will reject the null hypothesis.

The Statistical Test Suite developed by NIST is an excellent and exhaustive document looking at various aspects of randomness in a *long sequence of bits*. They have documented 15 statistical tests and in each test it adopted first a procedure to find the statistic of chi-square variation χ^2 of a particular parameter for the given bit sequence with that obtained from the theoretical studies of an identical sequence under the assumption of randomness. It then adopted a technique to transform the χ^2 data to

a randomness probability data, named as $P - value$.
Among these tests is found the Binary Matrix Rank Test which is documented in the following.

2 Binary Matrix Rank Test

The focus of the test is the rank of disjoint sub-matrices of the entire sequence. The purpose of this test is to check for linear dependence among fixed length sub-strings of the original sequence. The main idea is to construct matrices of successive zeroes and ones from the sequence, and check for linear dependence among the rows or columns of the constructed matrices. The deviation of the rank - or rank deficiency - of the matrices from a theoretically expected value gives the statistic of interest.

2.1 Mathematical fundamentals

As described in the NIST SP 800-22 [1] Section 3.5, the test is based on the result of Kovalenko [3] and also formulated by Marsaglia and Tsay [4], being a specification of one of the tests coming from the DIEHARD [2] battery of tests. The result states that the rank R of the $M \times Q$ (M rows, Q columns) random binary matrix takes values $r = 0, 1, 2, \ldots, m$ where $m \equiv min(M, Q)$ with probabilities

$$p_r = 2^{r(Q+M-r)-MQ} \prod_{i=0}^{r-1} \frac{(1-2^{i-Q})(1-2^{i-M})}{1-2^{i-r}}.$$

The probability values are fixed in the test suite code for $M = Q = 32$. The number M is then a parameter of this test, so that ideally $n = M^2N$, where N is the new "sample size" and n is the length of the bits sequence. In practice, values for M and N are chosen so that the discarded part of the string, $n - NM^2$, is fairly small.
In this case, the rational is that $p_M \approx 0.2888...$, $p_{M-1} \approx 0.5776...$, $p_{M-2} \approx 0.1284...$ and all other probabilities are very small (≤ 0.005) when $M \geq 10$.
After obtaining the N square matrices, the binary rank R_l, $l = 1, ..., N$ is calculated and evaluated for each one, in order to determine the frequencies F_M, F_{M-1} and $N - F_M - F_{M-1}$:

$$F_M = \#\{R_l = M\}, \quad F_{M-1} = \#\{R_l = M-1\}.$$

The reference distribution for the test statistic is a χ^2 distribution:

$$\chi^2 = \frac{(F_M - p_M N)^2}{p_M N} + \frac{(F_{M-1} - p_{M-1} N)^2}{p_{M-1} N} + \frac{(N - F_M - F_{M-1} - p_{M-2} N)^2}{p_{M-2} N}$$

The reported $P - value$ is $e^{-\chi^2(obs)/2}$. To be able to interpret the test, large values of $\chi^2(obs)$ indicate that the deviation of the rank distribution from that corresponding to a random sequence is significant. If the computed $P - value$ is < 0.01, then conclude that the sequence is non-random. Otherwise, conclude that the sequence is random.

2.2 Implementation

In this section we present the logical schema, as shown in Figure 1 and the test implementation, together with the observations.

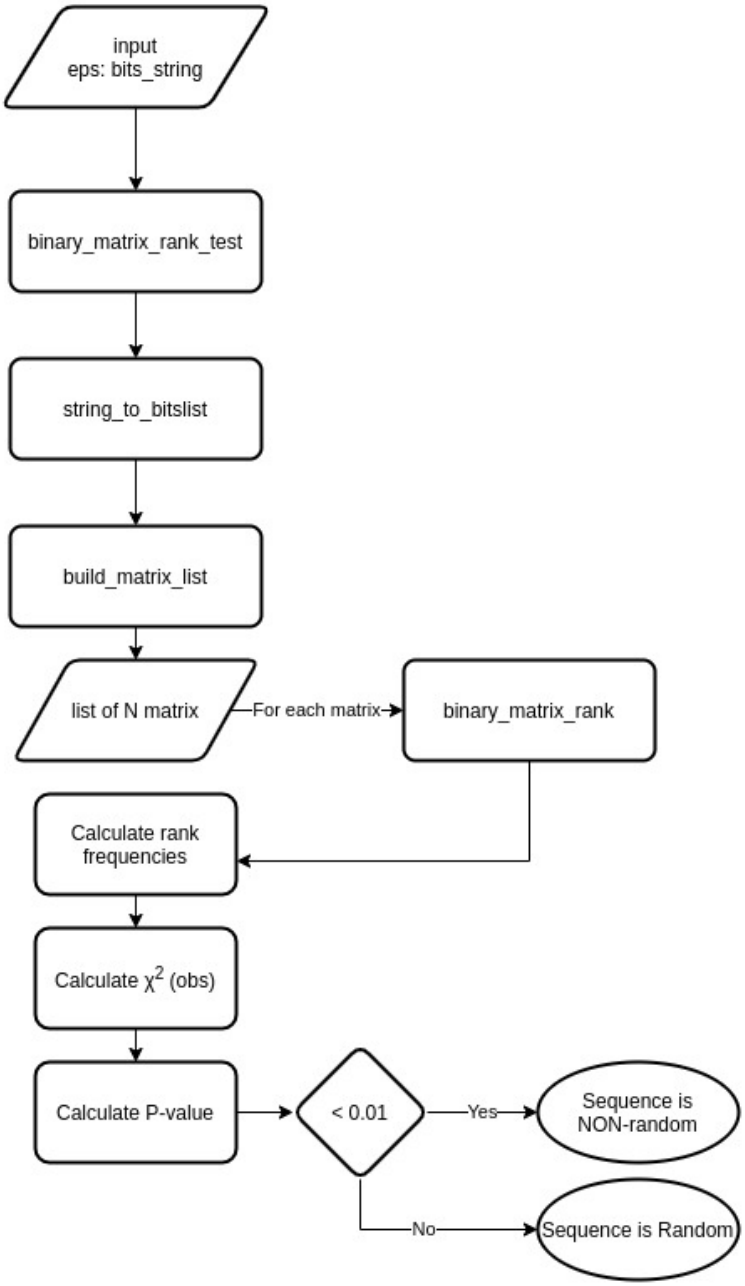

Figure 1: Logical schema

Following the above schema, the test algorithm takes as input a string of bits ϵ of length n and outputs the decision if the sequence is a random or not. The algorithm uses the fixed values for $M = Q = 32$ and their corresponding probabilities.

The first process converts the string of bits into a numeric (boolean) list of values, this being the preparation for creating and filling up the actual matrices. The next process does the actual building of every $M \times Q$ matrix, where the bits are inlined as usual from left to right line by line. The number of resulted matrices is $N = \lfloor \frac{n}{MQ} \rfloor$ and the left bits are discarded.

The third process iterates every matrix and computes the binary rank R_l for each one, where $l = 1, ..., N$.

The steps of computing the rank for a binary matrix are described in [nist] Appendix F.1. Further, the matrices with the rank M, $M-1$ are counted to obtain the frequencies F_M, F_{M-1} and $N - F_M - F_{M-1}$. The algorithm uses $M = Q = 32$, so it uses the probabilities $p_M = 0.2888$, $p_{M-1} = 0.5776$, $p_{M-2} = 0.1284$ as mentioned in the previous section, based on which the $\chi^2(obs)$ and the $P - value$ are calculated.
The last process implies the comparison of the $p - value$ with the threshold (0.01) so the assumption of randomness can be made.

Our solution is implemented in *Python* and can be found on *Github* at `https://github.com/Botoxit/NIST-BinaryMatrixRankTest`

3 Evaluation and observations

In this section we study the variations of the test outputs, like the variation of $P - value$ from density of bits.

As a first inputs sample, we used the random library from Python to generate bit strings of length $n = 100000$ and to analyse how the $P - value$ varies in relation to the density of the bits.
The test processed 2100 random generated inputs, from which only 18 were classified as *non-random* (around $0,857\%$). As shown in Figure 2 we observed that the density of the bits varies between $49 - 51\%$ of the data length.

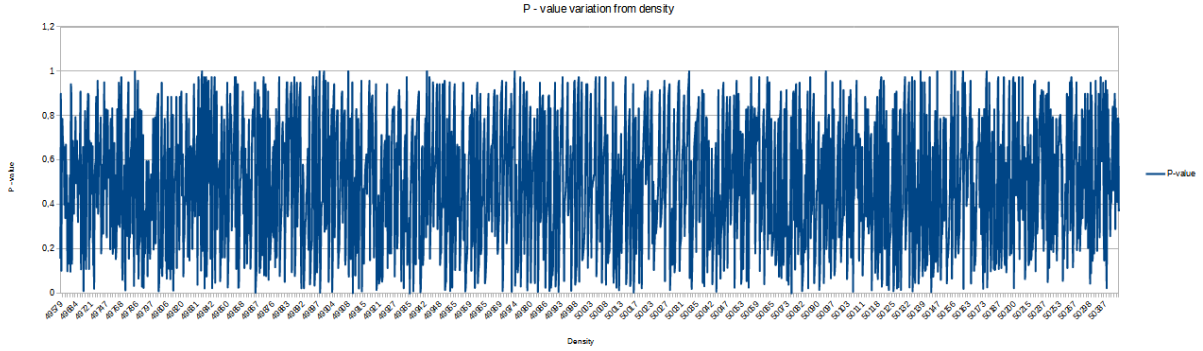

Figure 2: $P - value$ variation from bits density (Python random library)

For the next inputs sample, as shown in Figure 3, we used a combination between the Python random library and some alteration. We generated inputs of length 50000 having variations of the bits between 1 and 50000. We observe that we obtain $P - value$s greater than the threshold (sequence are random) for the density between $40 - 60\%$. The used sample length was 50000 so we could increase/decrease the variation with 1 on every test run. In this situation 41000 inputs are classified as *non-random* (around 82%).

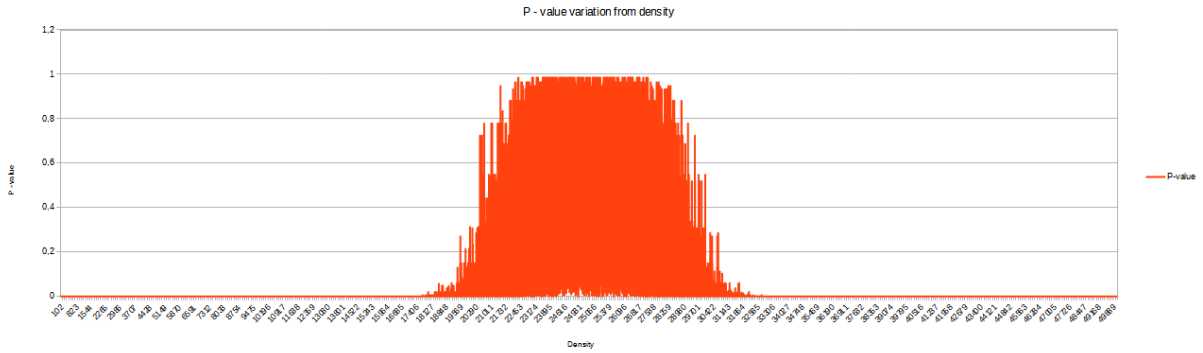

Figure 3: $P - value$ variation from bits density (density iterations)

Figure 4 shows a more narrow frame of the data from Figure 3, for the interval of density $40 - 60\%$.

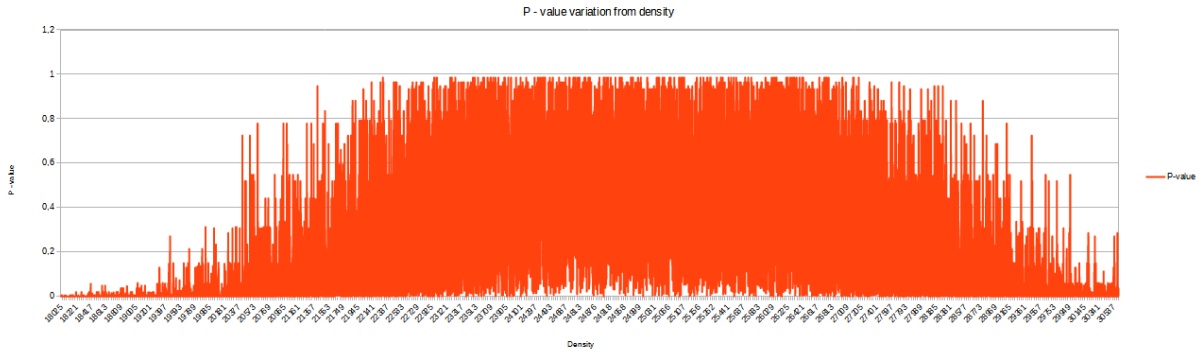

Figure 4: $P - value$ variation from bits density (density iterations $40 - 60\%$)

After we did the above test runs, there resulted ≈ 41000 inputs classified as *non-random* and ≈ 9000 inputs classified as *random* distributed over the density percentage as shown in Figure 5. We observed that, for a density around 50% we obtain a "peak" with more than 4000 random sequences - and almost no non-random sequences.

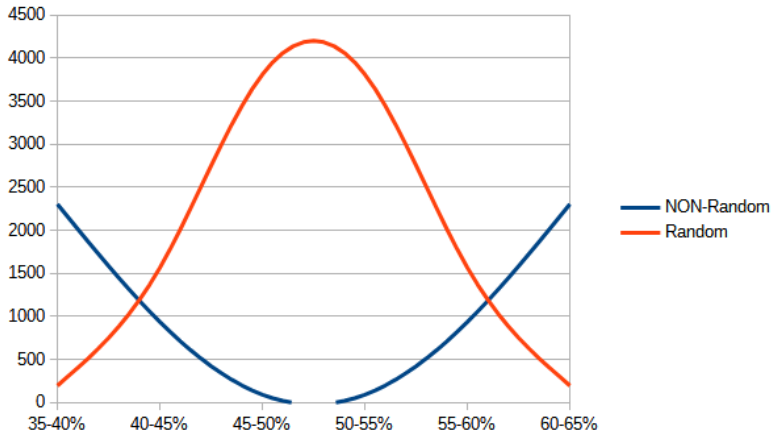

Figure 5: Random/Non-random count from density percentage

4 Conclusions

In this study we analyzed one of the NIST Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, more exactly the Binary Matrix Rank Test.

Beside the overview and the implementation of the test, we created some statistical experiments, then we made some remarks about the results. We concluded that a RNG or $PRNG$ should generate sequences of bits having the density as closed as possible to 50%. For a 50000 length sample, iterating all density possible values, we observed that, the more we went far from the 50% density , the more the random classified sequences number decreased.

References

[1] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J. and S. Vo, *"A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications"*, NIST 800-22, April 2010

[2] George Marsaglia, DIEHARD: a battery of tests of randomness. `http://www.stat.fsu.edu/pub/diehard/`.

[3] I. N. Kovalenko (1972), *"Distribution of the linear rank of a random matrix"*, Theory of Probability and its Applications. 17, pp. 342-346.

[4] G. Marsaglia and L. H. Tsay (1985), *"Matrices and the structure of random number sequences"*, Linear Algebra and its Applications. Vol. 67, pp. 147-156.