Evolutionary generation and degeneration of randomness to assess the indepedence of the Ent test battery

Julio Hernandez-Castro, David F. Barrero

***Abstract*—Randomness tests are a key tool to assess the quality of pseudo-random and true random (physical) number generators. They exploit some properties of random numbers to quantify to what extent the observed behavior of the tested sequence approximates the expected one. Given the many sides of randomness, there is not an unique test providing the whole picture, instead a suite of tests assessing different aspects randomness. A robust test suite must include independent tests, otherwise tests would assess the same property, providing redundant information. This paper addresses the independence assessment of a popular test suite named Ent. To this end we generate a large number of pseudo-random numbers with different degrees of randomness by evolving them with a Genetic Algorithm. The numbers are generated to maximize their diversity attending different criteria based on Ent output, used as fitness. We encourage diversity by maximizing and minimizing randomness measures. Once a diverse set of pseudo-random numbers is generated, the Ent test suite is run on them, and their statistics studied by means of a classical correlation analysis. The results show high correlation among some statistics used in the literature, which could be overestimating the quality of their randomness source.**

***Index Terms*—Randomness, randomness test, Genetic Algorithm, Ent, pseudo-random number generators, randomness sources.**

I. Introduction

Pseudo-random and real random number generation is a though problem with many critical applications. For instance, in Security almost any cryptographic system requires a reliable source of randomness in order to operate properly, which is a serious problem when dealing with deterministic machines such as a computer or a small IoT device. A weak pseudo-random number generator (PRNG) usually induces serious security vulnerabilities. This is why assessing the quality of those generators can have a major impact in Security. Any fail in generating good pseudo-random sequences may introduce weaknesses in a system.

Randomness tests are one of the best known tools to assess PRNGs. These tests verify some statistical properties of random numbers, and assess to which extend the sequence being tested satisfies those properties. The result of applying these tests is generally a quantitative evaluation, typically in the form of a p-value, of the quality of the PRNG. Given the importance of this topic, there is a number of well-known test suites, some of them even commercial. Perhaps the most used is the NIST test suite [1], [2], but others batteries such as Ent [3] or Crypt-X [4] are also widely known.

Ideally, different tests should assess completely different aspects of randomness. However, the complexity and subtle nature of randomness makes this far from trivial. Two conceptually different tests may well be, in essence, evaluating the same randomness characteristics and therefore producing correlated results. This is a serious threat for any test suite, because it might overestimate the quality of the assessed PRNG. In statistical terms, the independence of the evaluated properties translates to the independence of the statistical tests.

In this work we study the independence of the Ent randomness test suite. Ent is randomness test suite that includes five, supposedly independent, tests. To this end, we generate a sequence of pseudo-random numbers (PRN) with a strong PRNG, then apply a Genetic Algorithm (GA) to increase or decrease its randomness and then subject them the test battery. In this way we get a collection of various sequences with quite different statistical properties to avoid gaps. Once we got the test suite statistics, we look for correlations with classical statistical tools. Differently from previous studies, we generate a range of input pseudo-random sequences and analyze the test correlation based on their statistics, instead of the p-values. In this way we expect to increase the power of our analysis, by exploiting the extra information.

The paper is structured as follows. We first introduce related concepts, followed by a brief description of the Ent test battery. Section IV describes the research methodology in detail. The evolutionary generation of randomness as a maximization/minimization problem is described in section V. Section VI describes the obtained results and the paper finishes with some conclusions and future work.

II. Related work

To the best of our knowledge, there are no previous studies on the independence of the Ent test suite. In general, the independence of randomness tests is a topic that has attracted little research interest so far, which contrasts with its remarkable importance, both theoretically and practically. Most of the existing studies on the independence of randomness tests are focused on the NIST test battery, which is understandable given its relevance, because the NIST suite is the most used in areas such as Security and Simulation.

Some attention has been given to speeding-up NIST tests implementations, because of their important practical implications. Approaches to improve NIST performance range from the development of new optimized implementations [5] to

removing [6] or merging [7] tests after an independence analysis.

Other authors have proposed more general frameworks to assess the independence of statistical tests. Fan et al. proposed a methodology to assess randomness tests correlation [8] and applied that method to assess the independence of NIST SP 800-22, finding some interesting correlations. The proposed method computes the difference between the p-values coming from two tests and studies the resulting distribution. Liu and Lai studied the independence of the NIST tests with short sequences and employing conditional entropy [9].

Doğanaksoy et al. a systematically studied the NIST tests independence in a set of related publications [10], [11], [12]. They generated all binary sequences of length 20 and 30 bits, passed the NIST tests on them and measured the proportion of rejections and the correlation among them. They tried to estimate tests sensibility, defined as the variance of the tests when the sequences are subject of mathematical transformations.

III. The Ent test suite

The Ent randomness test suite was proposed and implemented by John Walker, who developed the tests for different platforms and released them with a free license. Ent contains five tests, and has a verbose output which shows a number of statistics and other related data, as can be seen in Fig. 1.

Ent has two operation modes: binary and byte, with the latter one enabled by default. In binary mode, Ent considers each bit in the sequence while in byte mode the sequence is divided in bytes. Depending on the operation mode, Ent computes and displays different statistics. The operation mode also determines the expected value of the tests, so the tests output must be interpreted accordingly. For instance, in binary mode Ent considers binary symbols, represented by 0 and 1, therefore the expected arithmetic mean for a binary sequence is close to 0.5. On the contrary, in byte mode, each symbol is composed by 8 bits, and Ent interprets them as integer numbers, so a random sequence has an expected arithmetic mean close to 127.5 (see Fig. 1). Given that byte mode is more popular in the literature, and it is the Ent default operation mode, in the following we restrict our discussion to this mode.

Ent implements a collection of five tests from different origins. In the following we review them.

- **Entropy**. This test computes the entropy of the sequence under examination, as defined in classical Information Theory [13]. A random sequence should be rich in entropy. In byte mode, the maximum theoretical entropy of a long sequence is 8 bits per byte, while in binary mode it is one bit per bit.
- **Chi-square tests**. This is a widely used test described by Knuth in his classical book The Art of Programming [14]. The chi-square test computes the frequency of the symbols, and compares it with the frequency expected in a uniform distribution. To perform this comparison the chisquare statistic is computed.
- **Arithmetic mean**. As the name suggests, this is just the arithmetic mean of the symbols in the sequence. The expected statistic value for a true random sequence is 0.5 in binary mode and 127.5 in byte mode.
- **Monte Carlo Value for Pi**. This test interprets the sequence as coordinates in a square and counts the number of points that fall into a circle inscribed within the square. This number is used to estimate the value of pi. A truly random sequence will approximate pi with good accuracy, while a non-random number is not expected to approximate pi well. The test uses the statistic pierror, which measures the percentage error in estimating pi.
- **Serial Correlation**. [14]. The serial correlation test computes the correlation between two consecutive symbols (bits or bytes) in the sequence. A good random sequence would have low correlation, very close to zero, while a bad random sequence would lead to higher absolute values.

Figure 1 shows a typical Ent output in byte mode. It can be seen that Ent generates several output values. Along with the statistics previously described, Ent outputs other values such as the optimum compression, or excess of the chi-square test. All these values are usually considered in the literature when assessing a PRNG, so we will consider them in this study.

In summary, we consider seven statistics computed by Ent: Entropy, excess, mean, pierror, chisquared, compression and correlation.

IV. Experimental procedure

Our goal is to assess the independence of the Ent test suite. To this end, we use an experimental approach divided in three steps: First, we create a dataset of PRNs with different degrees of randomness according to different criteria; secondly, we run the Ent rest suite over the PRNs set previously generated, storing the output statistics, and finally perform a correlation analysis over the Ent statistics[1].

The procedure followed to create the dataset is described in detail in the next section. In few words we run a Genetic Algorithm (GA) to maximize and minimize the randomness of a population of numbers, storing all the individuals. In this way we have a set of PRNs with different properties in an attempt to reduce the probability of committing bias.

The issue of which output should be considered deserves some attention. In statistical frequentist inference, tests are performed first by computing a statistic from the data, and then inferring a p-value which is interpreted. Therefore both, statistic and p-value can be considered as output of the tests. Previous studies have usually used p-values to perform the analysis [8], [10]. This approach has the advantage of using the same scale and interpretation of the variable under study, however potentially useful information implicit in the statistic is lost. In our opinion this potential loss of information justifies using the statistic instead of the p-value, and for that reason in the following we use directly the tests statistics without further transformation.

[1]All the code, scripts and datasets needed to reproduce these experiments are in https://atc1.aut.uah.es/~david/cec2017

```
Entropy = 4.385614 bits per byte.

Optimum compression would reduce the size
of this 20180 byte file by 45 percent.

Chi square distribution for 20180 samples is 1266758.61, and randomly
would exceed this value less than 0.01 percent of the times.

Arithmetic mean value of data bytes is 56.5619 (127.5 = random).
Monte Carlo value for Pi is 3.611061552 (error 14.94 percent).
Serial correlation coefficient is 0.568671 (totally uncorrelated = 0.0).
```

Fig. 1. Example of Ent default output in byte mode.

Once there is a collection of PRNs along with their statistics, we can perform a correlation analysis to find dependencies among the tests. This analysis is done using basic statistical tool such as correlation matrices and scatter plot matrices. More sophisticated methods can be used, however this is a preliminary study looking for basic relationships.

The procedure so far described is controlled by two variables: The sequence size and the direction of the evolution, i.e. evolve the random sequence towards more or less randomness. The sequence length is a basic factor to take into consideration since some statistics directly depend on this, like the Monte-Carlo method. To better describe the direction of evolution, we first should introduce in more detail the evolutionary procedure of generating the set of PRNs.

V. Generation and degeneration of random numbers with a GA

How to construct the set of PRNs is a relevant aspect which deserves some attention. Just generating a set of PRNs to test Ent relying on the underlying PRNG may bias the result. Randomness is a many-folded concept, far from being something monolithic. The PRNG would be creating PRN with some properties able to affect the tests. In order to tackle that concern, and try to get more robust results, we relied on a more sophisticated method to build the PRNs. In essence, the intention is to "randomize" the dataset of pseudorandom numbers.

In order to encourage diversity, we apply a GA codifying a population of random sequences. Other methods, such us using several PRNGs would be valid as well, but the proposed method based on a GA seems better suited to generate PRNs with diverse properties. The GA has a classical design, with binary codification, tournament selection, one-point crossover and bit flip mutation. Table I summarizes the most relevant GA settings. The initial population is generated randomly with a Twister-Mersenne PRNG, then the GA is run several times to maximize a randomness measure based on the Ent statistics used as fitness. All the individuals in the algorithm executions are stored for further analysis.

Table II shows the Ent statistics used in the experiments, along with the associated fitness function. For implementation simplicity we needed to express the fitness as a maximization problem, but some Ent statistics are inversely proportional to randomness, therefore raw statistics were not suitable as fitness. Table II shows the transformations done to the statistics to introduce them in the fitness functions. Fitness associated to entropy is the raw entropy, the same applies to excess. Compression is just the difference with the maximum compression, 100. Chisquared, pierror and correlation are inverted with a sum for safety. The interpretation of the fitness values is not

TABLE I
Genetic Algorithm parameters tableau.

Population	100
Length	Fixed to several values
Codification	Binary
Crossover	One-point crossover
Mutation	Bit flip mutation
Mutation probability	0.005
Selection	Tournament selection, size two
Elitism	1
Fitness	See Table II

TABLE II
Statistics of the Ent test suite and the fitness values used in the GA.

Ent statistic	Fitness value
Entropy	$f_{entropy} = Entropy$
Compression	$f_{compression} = 100 - Compression$
Chisquared	$f_{chisquared} = \frac{1}{1+Chisquared}$
Excess	$f_{excess} = Excess$
Average mean	$f_{amean} = \frac{1}{(1+\lvert 127.5-amean\rvert)}$ (byte) $f_{amean} = \frac{1}{(1+\lvert 0.5-amean\rvert)}$ (binary)
Pierror	$f_{pierror} = \frac{1}{1+pierror}$
Correlation	$f_{correlation} = \frac{1}{1+\lvert correlation\rvert}$

a concern, since we are only interested in creating PRNs with some diversity.

The PRNG used to populate the first generation of the GA is pretty good. If the GA only maximizes randomness, then the set of PRNs will only contain high randomness, with individuals in a tight range of fitness values. This is against the policy of encouraging diversity. This potential problem is easy to overcome, in addition to maximizing the randomness, we also degenerated it in a controlled way. This can be achieved changing the tournament selection to pick up the worse individuals instead of the best one, transforming the maximization problem into a minimization one.

VI. Experimental results

A. PRNs generation

We generated a dataset of PRNs with the method described controlling with two variables: Sequence length and fitness minimization (randomness generation)/minimization (randomness minimization). Each configuration was run five times for 50 generations. The result of randomness generation is depicted in Fig. 2 while the result of degenerating randomness is depicted in Fig. 3, both figures depict the average fitness versus the generation.

Figure 2 shows that regardless of the statistic used in the fitness function, the GA is able to increase the fitness. This is not an obvious result. The initial population was created with Twister-Mersenne, which is a cryptographycal-quality PRNG. It means that the initial population contains high quality PRNs, therefore their randomness should be difficult to improve. The increase of the average fitness (which also can be observed in the best fitness, not shown here) suggests that the GA is able to find even better PSNs. This can be observed for all the fitness functions and at least the smaller chromosome lengths.

Not all the fitness functions are equally successful increasing randomness, there are notable differences among them. For instance, when fitness is based on the chisquare, evolution is faster than compression. This is something we could expected, since compression is given by Ent as an integer number while the rest of statistics are floats with more precision. In other words, some statistics show more sensibility to changes in the individuals, yielding a more friendly fitness landscape which helps evolution.

Chromosome length has a strong influence in the evolution of randomness. As usual in GAs, longer chromosomes means larger search spaces, inducing more difficult searches. This can be seen in Fig. 2, where the fitness slope is smaller as the chromosome length increases. More interesting is the asymptotic behavior that the fitness seems to have, it is particularly clear looking at the entropy-based fitness in Fig. 2, where it seems to be bounded. As the chromosome length increases, the difference tends to dilute. This only happens in byte mode (data in binary mode not shown), and is related with the frequencies distribution, when the chromosome length is short, the frequencies distribution is unbalanced, and therefore some tests will interpret that like lack of randomness, affecting the statistic value.

Fig. 3 depicts the evolution of the average fitness with the number of generations when minimizing the fitness. Compared to maximization, reducing the randomness is easier than increasing it; fitness values evolve faster in minimization than in maximization in all the cases. This is not surprising, since we initialized the population with high quality PRNs. The notable influence of the chromosome length is also clear in minimization, with asymptotic behaviors dependent on the length.

Fitness values are more widely distributed in minimization (see Figs. 3) than maximization (Fig. 2), which is our objective. Altogether, maximizing and minimizing GAs, provide a wide range of PRN based on their fitness, ranging from high quality PRNs to non-random numbers.

The final dataset used for the correlation analysis was constructed by massively running the GA previously described. Given the relevance of the chromosome length, we generated numbers with different lengths. The fitness function attempted to maximize entropy, compression, chisquared, excess, average mean, pierror and correlation, as expressed in Table II. The GA was configurated to maximize or minimize the fitness function just manipulating the tournament selector.

B. Ent statistics analysis

We run Ent on the dataset generated by the GA in byte mode and collected its output to perform a correlation analysis. We divided this analysis into three steps: First we analyzed the effect of the chromosome length and then computed correlation matrices for the Ent tests statistics.

1) Effect of the length: Figure 4 shows the scatter matrix of the statistics for different chromosome sizes. For clarity, we only depicted small chromosome lengths; longer lengths follow the same pattern observed. The scatter plot matrix is quite interesting, showing a number of facts. .

The statistics present different degrees of sensibility towards the length. A good tool to visualize this sensibility is the density plots included in the diagonal of the scatter plot matrix in Fig. 4. These plots depict the empirical distribution of the statistics values grouped by the length. The distribution of entropy and compression is highly determined by the length, depending on it, means of entropy and compression take different values with very small variance. This behavior is observed when plotting these two statistics against any other. It is clear that sequence length changes the mean entropy and compression values, not affecting their variance. We should mention that the dataset does not contain very low quality PRN, otherwise the variance of entropy should be higher.

Arithmetic mean has the opposite behavior to entropy and compression. The mean of the arithmetic mean does not seem to change with the length, but its variance does, being smaller as the length increases. This behavior has an statistical explanation. This density plot shows the sampling distribution of a mean, and thus, by the Central Limit Theorem we must expect the distribution of the mean sample to have the same mean and standard deviation inversely proportional to the square root of the sample size (i.e. the length). The correlation

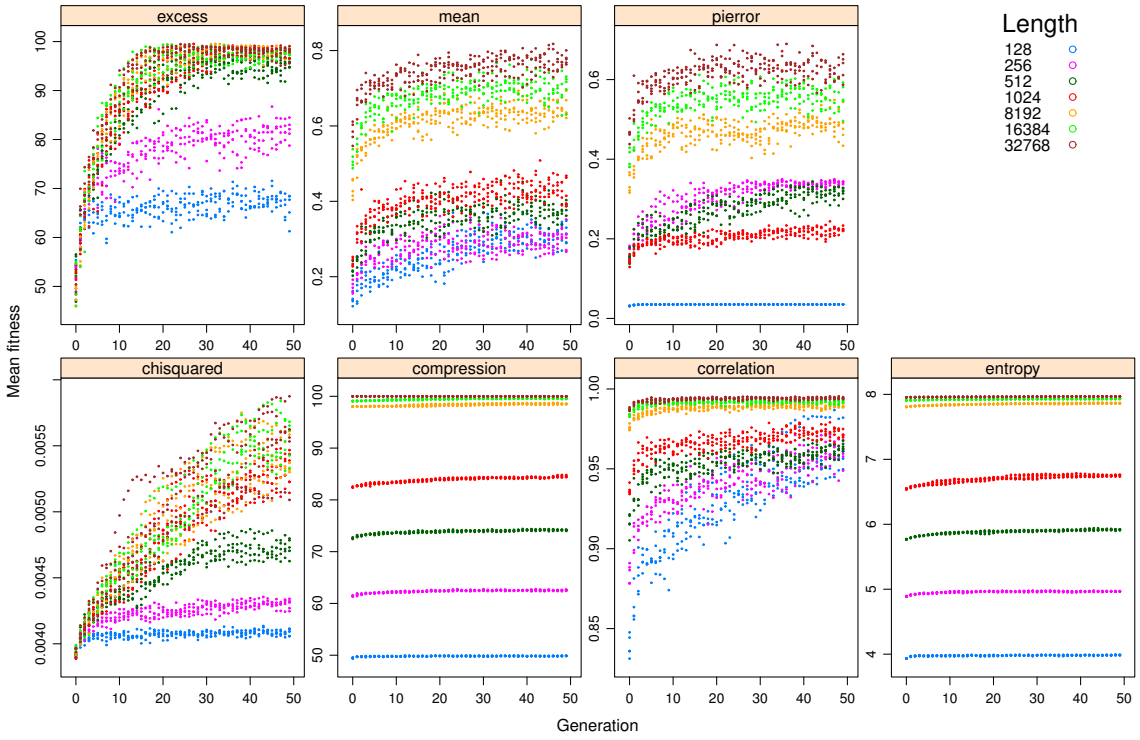

Fig. 2. Average fitness of the GA *maximizing* randomness according to different statistics for with different chromosome lengths. Length is expressed in bits. Corresponding fitness functions are in Table II.

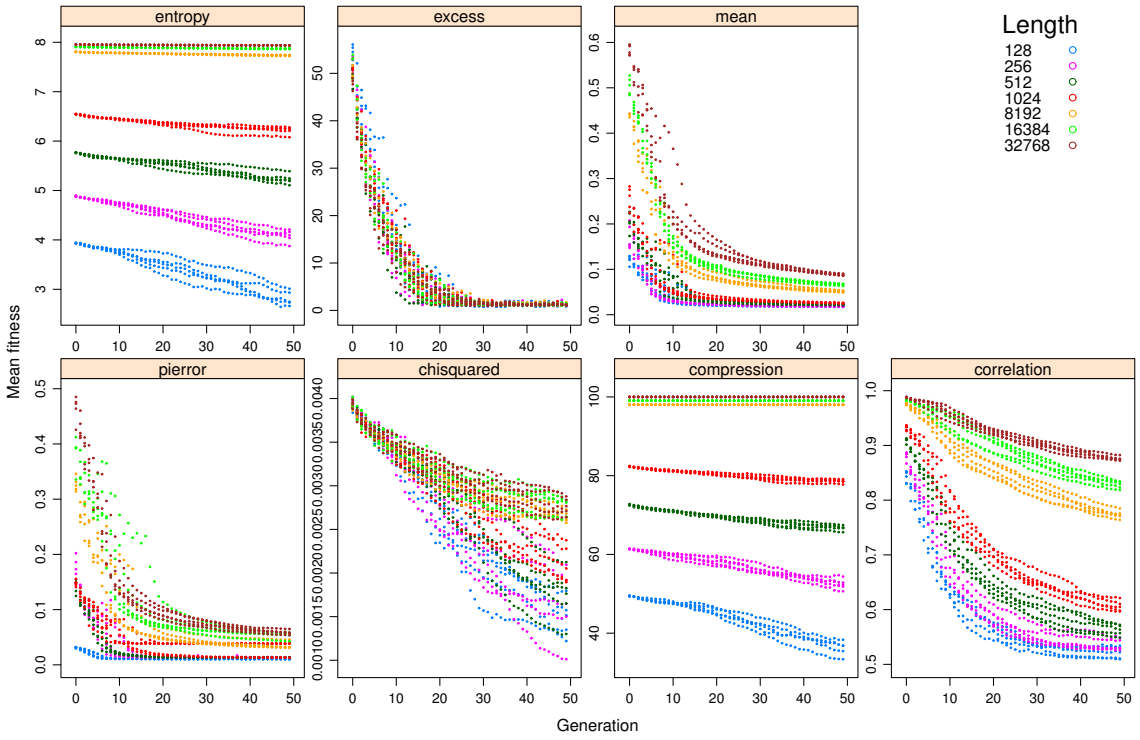

Fig. 3. Average fitness of the GA *minimizing* randomness according to different statistics for with different chromosome lengths. Length is expressed in bits. Corresponding fitness functions are in Table II.

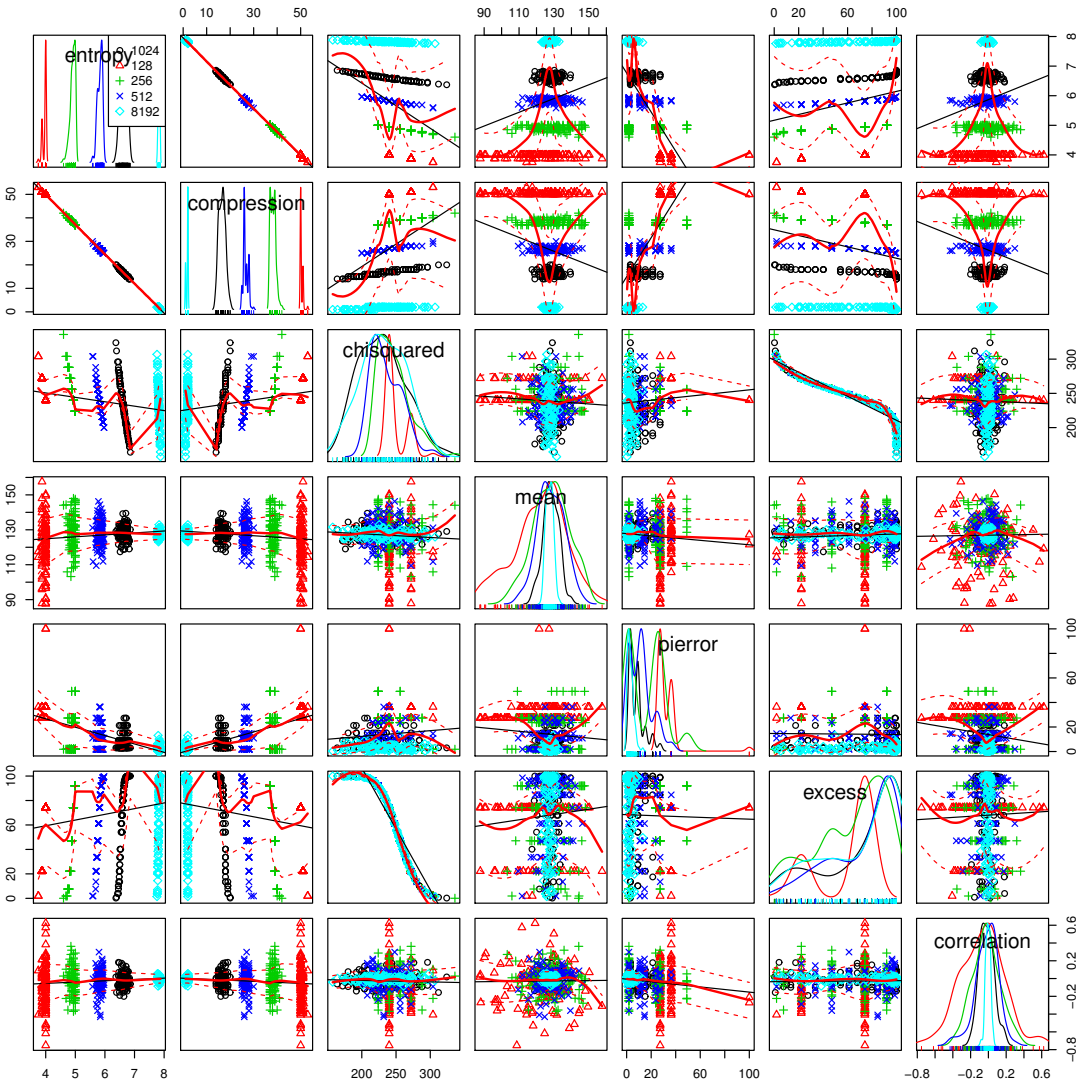

Fig. 4. Scatterplot matrix relating pairs of statistics grouped by sequence lengths (128, 256, 512, 1.024 and 8.192 bits). 15 samples for each pair objective-length. Longer chromosomes show the same pattern (data not shown). Dataset obtained with a GA maximizing the statistics, the same pattern is observed with minimization (data not shown).

seems to have a similar behavior, as it is computed as the average of several correlation coefficients. Pierror and excess do not exhibit a clear pattern when controlled by the length, while chisquared is affected by the length in a not evident way.

In any case, the inspection of Fig. 4 clearly shows that the length is a factor that must be taken into account. The lack of control of the length would rise to higher variance in the data, and that might hide correlation. This is particularly clear when the lengths are small; its influence seems to dilute as the length increases.

2) Correlation analysis: An inspection of Fig. 4 identifies some pretty clear correlations. In particular, the scatter plot of entropy and compression shows an almost perfect line, suggesting a strong relationship between them. Checking out the source code of Ent we can verify that compression is directly computed from the entropy, with a linear relationship broken by a ceiling operator. This is an important observation because some studies use entropy and compression as independent statistics while they are not.

A superficial analysis of the correlations suggested by Fig. 4 may be deceptive. Entropy and compression seem to by highly correlated with all the other statistics, at least the points on the plane follow a linear pattern. If we consider the length, we appreciate that there is a scale effect, more significant with large values of length. This is pretty clear, for instance, in the entropy-mean scatter plot. The variance of the mean is affected by the length, while the entropy variance is not; as a consequence, the ratio width-height tends

to one as chromosome length increases. In few words, the correlation in this case decreases with the length. This fact is consistent with the way in which randomness tests are used, they are typically applied to long sequences. However, many independence studies use small lengths; out results suggest that their conclusions may not generalizable to larger sequences.

Table III contains the Pearson's correlation coefficients of each pair of statistics, for sequences length equal to 8192 bits and PRNs generated by randomness maximization and minimization. For clarity, those coefficients higher than 0.75 are marked in bold. This correlation matrix is representative of larger sequence lengths. Both GA strategies, randomness maximization and minimization achieve comparable results, as can be seen in the table. Entropy gets correlations higher than 0.75 with compression, chisquared and excess. All these correlations are visually identified looking at Fig. 4.

It is perhaps surprising to find a correlation as low as 0.75 between entropy and compression when we know from the Ent source code that they have a linear relationship, and it appears almost linear in the scatter plot. A closer look to the plot may explain this. Fig. 5 shows another scatter plot matrix restricted to lengths equal to 8192 bits. The density plot of compression reflects its discrete nature; compression is given by Ent as an integer number, so compression only takes two values, two and three. When the compression is paired with entropy to draw the scatter plot, the result is a step function. This fact may be degrading the correlation coefficient underestimating their true correlation, which should be close to one.

Entropy is highly correlated with chisquared, which in turn is also correlated with compression. This is a consistent result, there is a chain of correlations. Entropy is also highly correlated with excess, so we should expect that compression should also be highly correlated with excess, however, it is not clear. The correlation coefficient between compression and excess is -0.616 (maximization) and -0.521 (minimization). This is a notable correlation, but quite lower than expected. The discrete values of compression (see Fig. 5) explains it.

VII. Conclusions

Independence of randomness statistical tests is a relevant problem that has attracted little research interest. In this paper we have analyzed the independence of a popular randomness test suite implemented in a software package named Ent. On the contrary that other studies, we based our analysis on the tests statistics instead of the p-values, in this way we try to maintain all the information that is lost in the p-values computation.

In first place we generated a dataset of random sequences. The straightforward way to do this is just using a PRNG, but this might bias the result. We encouraged diversity by taking the output of a PRNG and using a GA to increase and decrease in a controlled way the quality of the randomness, understanding by quality the statistics used by the Ent test suite. Then we computed the Ent statistics for the dataset and applied a basic correlation analysis to verify their independence. In total we evaluated seven statistics (entropy, compression, chisquared, arithmetic mean, pierror, excess and correlation) associated to five tests.

The results show that the studied statistics are not completely independent. Entropy and compression are almost the same measure, with a linear transformation and a ceilling operation, which actually makes compression in Ent, in our opinion, a bad choice to assess randomness, so we suggest not using it. Chisquared and excess also show high correlation, therefore their information is redundant, it should be safe to only use one of them. In summary, the evidences shown here suggest that the seven analyzed statistics could be reduced to five without great loss of information.

The correlation analysis performed in this study used some basic tools. A future research line is including more powerful methods to assess the tests independence. In this sense Machine Learning provides tools, such as PCA or regression analysis, that we expect to apply in a near future.

Acknowledgment

This work has been supported by UAH project (2016/00351/001), UAH Mobility Grant and Spanish Ministry of Economy and Competitivity project EphemeCH (TIN2014-56494-C4-4-P).

References

[1] A. Rukhin, J. Soto, J. Nechvatal, S. Miles, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," p. 131, 2010.

[2] J. Soto, "Statistical testing of random number generators," in *Proceedings of the 22nd National Information Systems Security Conference*, vol. 10, no. 99. NIST Gaithersburg, MD, 1999, p. 12.

[3] J. Walker, "Ent: A pseudorandom number sequence test program," *Software and documentation*, 2008. [Online]. Available: http://www.fourmilab.ch/random/

[4] H. Gustafson, E. Dawson, L. Nielsen, and W. Caelli, "A computer package for measuring the strength of encryption algorithms," *Computers & Security*, vol. 13, no. 8, pp. 687 – 697, 1994.

[5] M. Sýs and Z. Říha, "Faster Randomness Testing with the NIST Statistical Test Suite." Springer International Publishing, 2014, pp. 272–284.

[6] P. Hellekalek and S. Wegenkittl, "Empirical evidence concerning AES," *ACM Trans. Model. Comput. Simul.*, vol. 13, no. 4, pp. 322–333, 10 2003.

[7] M. Sýs and V. Matyáš, "Randomness Testing: Result Interpretation and Speed." Springer Berlin Heidelberg, 2016, pp. 389–395.

[8] L. Fan, H. Chen, and S. Gao, "A General Method to Evaluate the Correlation of Randomness Tests," in *Information Security Applications*. Springer International Publishing, 2014, pp. 52–62.

[9] C. Liu and X. Lai, "Evaluation of Statistical Tests for Randomness Using Conditional Entropy," in *2013 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*. IEEE, 10 2013, pp. 504–509.

[10] M. Sönmez Turan, A. Doğanaksoy, and S. Boztaş, "On Independence and Sensitivity of Statistical Randomness Tests," in *Sequences and Their Applications - SETA 2008*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 18–29.

[11] A. Doğanaksoy, B. Ege, and K. Muş, "Extended results for independence and sensitivity of NIST randomness tests," in *Information Security and Cryptography Conference, ISCTurkey*, 2008.

[12] 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 & Computer Sciences*.

[13] R. W. Hamming, *Coding and Information Theory*. Prentice-Hall, 1980.

[14] D. E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley Professional, 2005.

TABLE III

PEARSON'S CORRELATION MATRIX RESTRICTED TO CHROMOSOMES WITH LENGTH 8192 BITS. THE LEFT VALUE IS FOR MAXIMIZATION WHILE THE RIGHT VALUE IS FOR MINIMIZATION. VALUES LARGER THAN 0.7 ARE MARKED IN BOLD, VALUES BETWEEN 0.5 AND 0.7 ARE IN ITALICS.

	Entropy	Compression	Chisquared	Mean	Pierror	Excess	Correlation
Entropy	1.0	**-0.796 / -0.811**	**-0.979 / -0.976**	0.111 / -0.119	0.060 / 0.069	**0.902 / 0.835**	-0.091 / -0.079
Compression	-	1.0	**0.784 / 0.800**	-0.109 / 0.089	-0.052 / -0.071	*-0.616 / -0.521*	0.091 / 0.065
Chisquared	-	-	1.0	-0.109 / 0.115	-0.060 / -0.075	**-0.925**, **-0.837**	0.093 / 0.083
Mean	-	-	-	1.0	-0.002 / 0.319	0.094 / -0.099	-0.011 / -0.021
Pierror	-	-	-	-	1.0	0.051 / 0.045	0.030 / -0.018
Excess	-	-	-	-	-	1.0	-0.088 / -0.083
Correlation	-	-	-	-	-	-	1.0

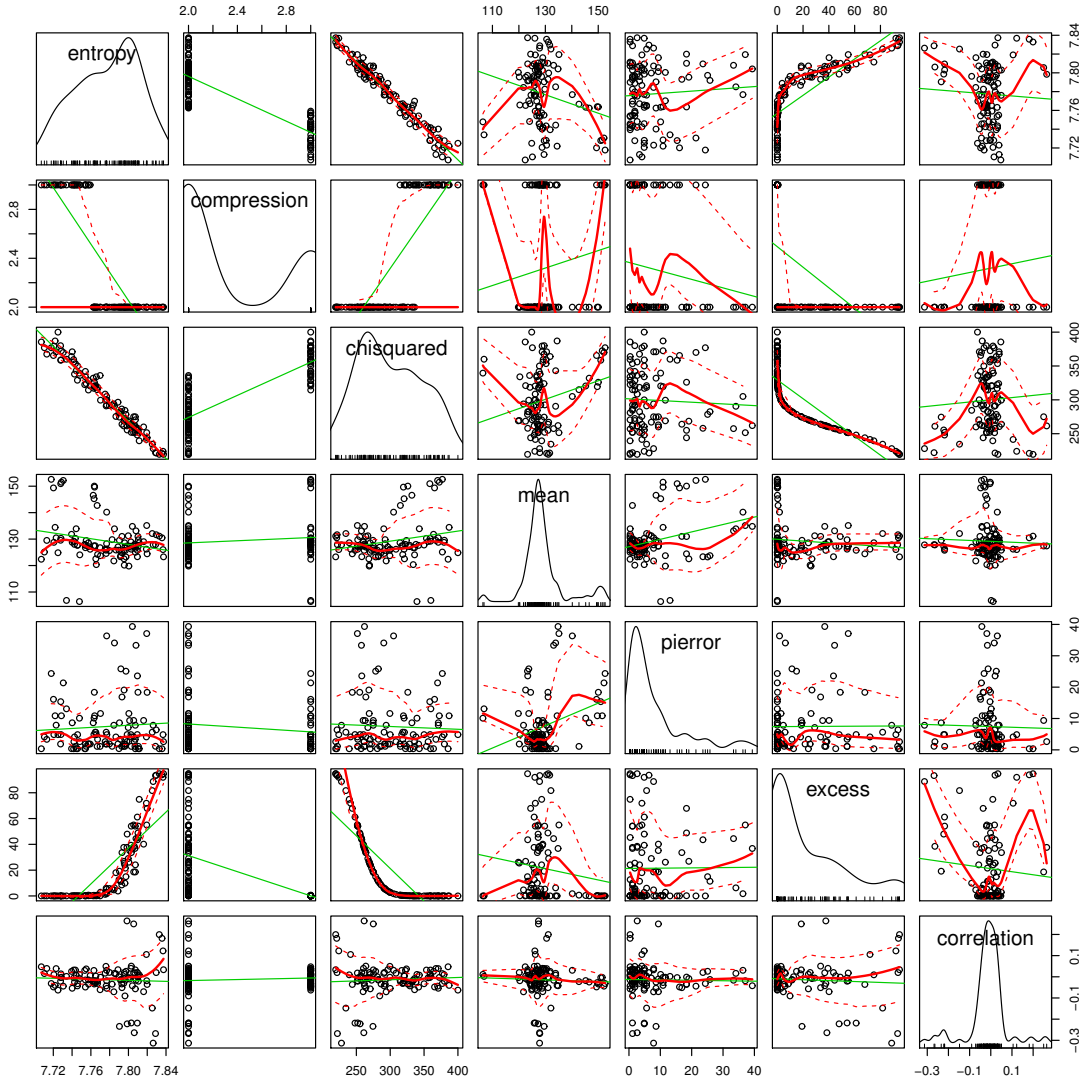

Fig. 5. Scatterplot matrix for randomness maximization and sequence lengths of 8192 bits.