UNIVERSITÀ
DI TORINO

Politecnico
di Torino

Doctoral Dissertation
Doctoral Program in Pure and Applied Mathematics (35^{th}cycle)

Randomness Tests for Binary Sequences

By

Guglielmo Morgari

******

Supervisor:
Prof. Danilo Bazzanella

Doctoral Examination Committee:
Prof. Andrea Visconti, Referee, Università di Milano
Prof. Nadir Murru, Referee, Università di Trento
Prof. Antonio José Di Scala, Politecnico di Torino

Università di Torino - Politecnico di Torino
2023

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Guglielmo Morgari
2023

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the doctoral programme jointly established by Politecnico di Torino and Università degli Studi di Torino

Dedicated to the memory of
Professor Emeritus Michele Elia,
dear Friend and enlightening Mentor

Acknowledgements

It has been a long, tough, and deeply rewarding journey, and if I have reached the end, I owe many people my deepest gratitude. The list of those I wish to thank is long.

Michele Elia, for honoring me with his friendship for many years and for generously sharing a wealth of knowledge in various fields of study and life.

Danilo Bazzanella, and the whole CrypTo group, for providing me with the opportunity to work in an inspiring, high-quality, respectful, and harmonious environment.

Andrea Visconti and Nadir Murru, for their meticulous review of my work and for providing numerous invaluable suggestions for its enhancement.

Antonio José Di Scala, for the few and occasional but always illuminating meetings we have had over the years.

Umberto Cerruti, for sparking my passion for mathematics many years ago with his fascinating Algebra courses.

Vittorio Bagini, for patiently reading my work with impressive and meticulous persistence, allowing me to eliminate many errors and give the work a more solid and coherent structure.

Franco Maddaleno, for teaching me how to mentally calculate the square of a number ending in 5, when I was a child, among many other valuable lessons.

Marco Coppola, the Socio, with whom I shared many years of work and who has always been an unparalleled and enjoyable source of inspiration and exciting discussions.

Fabrizio Vacca, whose encouragement was decisive in convincing me to embark on this challenging journey a few years ago.

Edoardo Signorini, Francesco Stocco, Giuseppe D'Alconzo, Giuseppe Laurenza, Marco Rinaudo, and Veronica Cristiano, with whom I have the privilege to work every day, for their patient and friendly support throughout my whole project.

Alessandro Giacchetto for kindly sharing some results from his Master's thesis.

Telsy, the company where I have spent the majority of my (now quite long) professional life, for offering me the opportunity to delve into the fascinating subject of randomness and reason about it, exchanging ideas with many talented friends and colleagues.

My mother, my sisters, my family-in-law, and all my close friends who have always encouraged me over time, providing me with the serenity and the confidence necessary to complete my work.

My father, with great regret for not being able to share with him the satisfaction of this achievement.

Lastly, I am deeply and joyfully indebted with my beloved Chiara and Alice, for their boundless patience, care, and support during this long journey. Without their invaluable help, reaching the end would have been simply impossible.

Abstract

The generation of random numbers
is too important to be left to chance[1]

Robert R. Coveyou,
Oak Ridge National Laboratory

Cryptography is vital in securing sensitive information and maintaining privacy in the today's digital world. Though sometimes underestimated, randomness plays a key role in cryptography, generating unpredictable keys and other related material. Hence, high-quality random number generators are a crucial element in building a secure cryptographic system. In dealing with randomness, two key capabilities are essential. First, creating strong random generators, that is, systems able to produce unpredictable and statistically independent numbers. Second, constructing validation systems to verify the quality of the generators.

In this dissertation, we focus on the second capability, specifically analyzing the concept of *hypothesis test*, a statistical inference model representing a basic tool for the statistical characterization of random processes. In the hypothesis testing framework, a central idea is the *p-value*, a numerical measure assigned to each sample generated from the random process under analysis, allowing to assess the plausibility of a hypothesis, usually referred to as the *null hypothesis*, about the random process on the basis of the observed data.

P-values are determined by the probability distribution associated with the null hypothesis. In the context of random number generators, this distribution is inherently discrete but in the literature it is commonly approximated by continuous distributions for ease of handling. However, analyzing in detail the discrete setting, we show that the mentioned approximation can lead to errors. As an example, we thoroughly

[1]Thanks to Antonio José Di Scala for pointing out this brilliant quote.

examine the testing strategy for random number generators proposed by the National Institute of Standards and Technology (NIST) and demonstrate some inaccuracies in the suggested approach. Motivated by this finding, we define a new simple hypothesis test as a use case to propose and validate a methodology for assessing the definition and implementation correctness of hypothesis tests. Additionally, we present an abstract analysis of the hypothesis test model, which proves valuable in providing a more accurate conceptual framework within the discrete setting.

We believe that the results presented in this dissertation can contribute to a better understanding of how hypothesis tests operate in discrete cases, such as analyzing random number generators. In the demanding field of cryptography, even slight discrepancies between the expected and actual behavior of random generators can, in fact, have significant implications for data security.

Contents

List of Figures

List of Tables

List of Acronyms

AES Advanced Encryption Standard

CDF Cumulative Distribution Function

LCG Linear Congruential Generator

LFSR Linear Feedback Shift Register

GCM Galois Counter Mode

NIST National Institute of Standards and Technology

NIST-STS NIST Statistical Test Suite

OEIS On-Line Encyclopedia of Integer Sequences

PDF Probability Density Function

PMF Probability Mass Function

PRNG Pseudo Random Number Generator

TRNG True Random Number Generator

Chapter 1

Introduction

The aim of this dissertation is to provide a contribution to the theory of statistical analysis of random numbers generators. In particular the scenario considered is that of generators producing binary sequences, especially for cryptographic applications which, by their nature, are very demanding in terms of randomness quality.

The current chapter contains a brief introduction to the topic addressed in this dissertation and is organized as follows: in §1.1 a quick overview of the subtle relation between cryptography and randomness is given, followed in §1.2 by some examples of cryptographic failures due to misuse of randomness. Then in §1.3 a brief description of the two main classes of random generators (physical and logical devices able to produce (pseudo)randomness) is given and the National Institute of Standards and Technology (NIST) recommendations to build and validate random generators are introduced. In the following §1.4 some considerations on how to test the quality of a random generator are reported. Finally, in §1.5, the organization of the chapters of this dissertation is given.

1.1 Cryptography and Randomness

Cryptography and randomness are closely related concepts, even if this relationship is often underestimated or not fully understood. Randomness is, in fact, an absolutely fundamental and essential element in the security of a cryptographic system and represents a critical link in every information security chain. Misuse of random numbers almost invariably results in vulnerable cryptography in applications.

A cryptographic system is usually described in terms of algorithms and protocols, that is, of deterministic primitives, assuming that the choice of robust (deterministic) components implies the security of the resulting system. In reality, the strength of these components is typically implicitly based on the existence of secret material (keys and other cryptographic parameters), which is supposed unknown to anyone but the legitimate users. If this material were obtainable by an attacker, then the security of the entire system would be compromised. Hence, the need to be able to generate high-quality random material, which the attacker cannot guess (of course an attacker might have non-cryptographic means to recover the secret information, like threatening or bribing, but this is obviously outside the scope of this work).

The use of random generators impacts the very basic tools and mechanisms of cryptography including, among others, encryption, message authentication, digital signatures, PIN and password generation.

1.2 Examples of cryptographic systems failures

Without going into the details and the exact meaning of the mechanisms listed above, we point out that a poor understanding of the importance of randomness in cryptographic systems often leads to catastrophic compromises of the resulting security.

This section contains some relevant examples of cryptographic system failure caused by poorly implemented randomness, but many others can be easily found in literature.

1.2.1 Netscape SSL implementation

A well-known case concerns the first implementations of the Secure Socket Layer (SSL) protocol by Netscape, in which the production of cryptographic material relied on a deterministic algorithm (pseudo random noise generator, see §1.3.2) initialized with the value of three system-related variables: the time of day, the process ID, and the parent process ID.

Although apparently these three values have a good variability, in fact they result in low entropy[1] allowing an attacker to predict the cryptographic parameters produced and thus, ultimately, break the system. Given the widespread use of SSL it is easy to understand the devastating impact of the mentioned attack, proposed by Goldman and Wagner in 1996 [1].

1.2.2 Debian random generator

Another notorious example concerns the random number generator implemented in the version of OpenSSL present on Debian Linux and other Debian-based distributions. Due to an implementation error, introduced in 2006, the resulting entropy of the generator was much lower than expected, making the keys produced by the generator potentially vulnerable to brute force attacks. The problem was very important because it affected keys used in extremely popular systems, such as SSH, OpenVPN, and TLS. In 2008 the error was detected and corrected and many weak keys replaced, but it is likely that many other weak keys are today still in use. See [2] for the security advisory document released by Debian.

1.2.3 RSA public keys factoring attack

In 2012 Lenstra, Hughes, Augier, Bos, Kleinjung, and Wachter ([3]) collected millions of RSA public keys from X.509 certificates and PGP keys available on the Internet[2] . They found that, in a surprisingly large fraction of these keys, the RSA modules shared a prime factor with other keys. With a simple application of Euclid's algorithm this allowed to factor the modules and then break the RSA keys. The reason for this weakness lies in poor quality random generators used to produce the keys, leading to output repetition.

[1] Entropy is the property of a random process which in some way formalizes and measures its unpredictability. A low-entropy secret is easy to guess by an attacker, typically through a *brute force attack*. Although it is here not necessary to go into the detail of the definition of entropy, it is reported in §A.6 for completeness.

[2] RSA [4] is a widely used cryptographic algorithm whose strength relies on the difficulty of factoring large numbers (under proper assumptions). X.509 [5, 6] is a standard for certificates used to securely deliver RSA keys. PGP [7] is a well-known encryption program using RSA keys.

1.3 Random number generators

Hence, if we want to build a cryptographic system, we must necessarily rely on the existence of high-quality random bit generators (or more generally random number generators), expected to be indistinguishable from ideal random generators. Unfortunately, generating random material (in fact bits) is a very slippery and in no way obvious task. The two basic classes of random generators, which in practice are often combined appropriately, are described in §1.3.1 and §1.3.2. The remarkable effort made by NIST, providing recommendations to build and then to validate random generators, is reported in §1.3.3.

1.3.1 True Random Number Generator (TRNG)s

TRNGs are based on the measurement of a given physical process considered to be random, like thermal noise or quantum phenomena. TRNGs inherently contain *true* randomness but often fail in terms of output production speed and quality. For example, they frequently contain some form of bias (e.g. prevalence of 1 over 0 or vice versa), which typically requires some post-processing to be removed. Furthermore, they are often quite slow. See for example [8].

1.3.2 Pseudo Random Number Generator (PRNG)s

At the other end of the spectrum are the PRNGs, consisting of a function which deterministically expands an initial value supplied as an external input (seed). The output of the expansion consists of long sequences entirely determined by the seed and therefore has little *true* randomness; however, if the expansion function is chosen appropriately, the process is very rapid and the sequences produced *seem* highly random. Hence they are commonly used in cryptographic applications[3]. See [9] for an overview.

[3]Of course the actual randomness is entirely given by the (short) seed and this must be considered when designing and analysing the application scenario.

1.3.3 NIST recommendations

Given the large quantity and heterogeneity of random generators proposed by academic researchers and industries, the NIST has produced an interesting series of recommendations in order to make their design easier and their evaluation more consistent. In this remarkable work NIST has clearly defined some fundamental concepts and provided useful guidelines in the construction of random number generators for cryptographic and non-cryptographic uses. In particular, NIST has released three Special Publications (SP).

In SP800-90A ([10]) the construction of different families of deterministic generators (PRNG) is considered. All these families are based on cryptographic techniques which, starting from an initial seed, generate sequences of bits with good statistical properties. SP800-90B ([11]) provides recommendations on the principles of design and validation of the entropy sources (in fact, TRNGs) used to supply the seeds to the PRNGs. The concept of entropy[4] is critical in a random generation system, as it formalizes and specifies the rather vague, imprecise and therefore slippery concepts of unpredictability and randomness. Finally, SP800-90C specifies how to implement random number generators by putting together the PRNGs defined in [10] and the entropy sources considered in [11]. In practice, the two approaches are often combined in order to obtain generation efficiency and high randomness: the TRNG is, in fact, typically used to produce the seed provided in input to the PRNGs. NIST's work is of considerable importance because it provides a solid reference framework for designing and analyzing random number generators, which, as shown, are a fundamental element in any information security system.

1.4 Quality assessment

In accordance with the above considerations, in order to gain confidence in the security of a cryptographic system it is necessary to assess the quality of the random number generator(s) used.

Gaining access to the design and internal details of the generator would obviously be of great help in its evaluation. However, very often this is not possible and the evaluation of the quality of the generator must be done indirectly by analyzing the

[4] For a formal definition of entropy see §A.6.

sequences it produces. In this dissertation we are going to focus on this second and more common scenario.

Measuring the quality of a random number generator through the analysis of the output sequences is a subtle and complex task. Defining an appropriate metric is, in fact, difficult and quite subjective and the problem can be approached from multiple point of views, which are often connected in non-obvious ways, including:

- estimating the unpredictability of the generator, through the concept of entropy;
- studying its algebraic properties (especially for PRNGs);
- analyzing the statistical behaviour of the output sequences.

A common methodology is to combine different approaches, since carrying out multiple and different analyses of a generator can certainly give good confidence on its quality (especially in a negative sense, when a "non-random" behavior is highlighted). However, it should be noted that no analysis, statistical or of any other nature, can give a definitive and absolute answer on the degree of randomness of a generator. This is not a technological but a conceptual limitation and has to do with the ambiguous and elusive nature of the very concept of randomness.

1.5 Dissertation structure

As anticipated at the beginning of Chapter 1, the goal of this work is to provide some contribution to the framework of random generators statistical analysis. The dissertation is structured in five more chapters (followed by the conclusions and an appendix), where Chapter 2 contains a description of the existing theoretical framework, while Chapters 3, 4, 5 and 6 constitute the original contribution of this work.

In particular, Chapter 2 gives a reasoned overview of the well-known *hypothesis test* model, a statistical inference model representing a basic tool for the statistical characterization of random processes. The main concept in the hypothesis test framework is the *p-value*, a numeric index associated to each sample produced by the random process, allowing to assess the plausibility of a hypothesis (typically said *null hypothesis*) about the random process on the basis of the observed data.

The setting considered in this chapter is very general and addresses primarily the continuous data distribution case, since the discrete case can often be approximated with a continuous one, typically much easier to manage.

However, in Chapter 3 it is observed that the above-described approximation can introduce subtle discrepancies between the expected and the observed behaviours, which in turn can result in wrong assumptions and, ultimately, in incorrect conclusions. The chapter is thus devoted to the detailed analysis of the discrete case and in particular considers the distribution of the p-values associated to a given test, as the sample varies in the sample space. The notion of *p-tuple* is defined as the tuple of p-values associated to a given test, which is then characterized in many respects and under different assumptions.

The following Chapter 4 proposes a more abstract definition of hypothesis test in the discrete case, based on the observation that the ultimate goal of a test is to differentiate between samples that provide evidence supporting the acceptance of the null hypothesis and those that result in rejecting it, while all the typical concepts associated to a hypothesis test, including the p-value, are in fact just intermediate tools. Hence, a test can also be seen as a partition of the sample space in two subsets, known as the acceptance region and the rejection region. Based on this view, a number of considerations is proposed.

Chapter 5 considers the most commonly used collections of statistical tests to assess the quality of random numbers generators. Among them, the most important is the NIST Statistical Test Suite (NIST-STS) which represents the *de facto* standard in the field and consists of 15 different tests. As for every collection of tests, a critical point is how to draw an overall conclusion on the basis of the results of the individual tests. Relying on the theory developed in Chapters 3 and 4, the methodology proposed by NIST is analysed, observing that some assumptions appear somewhat questionable.

Then, in Chapter 6, a toy hypothesis test for binary sequences is presented, with a rigorous analysis of its properties. Interestingly, despite its quite simple definition, the test turns out to be very effective against a specific well-known and widely used class of random generators, namely the Linear Congruential Generator (LCG)s. In this specific case the proposed test appears, in fact, more effective than all the common suites, including the one proposed by NIST.

In the following Chapter 7 the conclusions on the entire work are presented.

Finally, in Appendix A, the basic concepts of Statistics, Information Theory and Cryptography used throughout the dissertation are briefly reported.

Chapter 2

Hypothesis Tests

This chapter contains an introduction to the well-known hypothesis testing model The content is not meant to be exhaustive, rather it is focused on providing the theoretical framework necessary for the development of subsequent chapters. For a more detailed treatment of the topic, we refer to a probability book, like for example [12] or [13], §8.

Statistical analysis of a random generator is typically done by applying one or more *hypothesis tests* to the sequences produced by the generator. *Hypothesis tests* are a general class of statistical tests commonly used in a very broad range of application scenarios, for which we want to verify a certain hypothesis on the basis of the observations of available experimental data (samples). The output of these tests provides an indication of the consistency of the observed data with the given hypothesis.

The chapter is organized in two main sections. In §2.1 we summarize how a generic hypothesis test works, introducing the main related concepts. In §2.2 we describe and compare the two methods typically used in its implementation (namely the *critical value method* and the *p-value method*).

2.1 Hypothesis test scheme

In the general scheme of a *hypothesis test* we consider a random experiment, where samples are randomly extracted from a given sample space. According to the goal

of the test, two hypotheses are defined in terms of the underlying distribution of the sample extraction process, also known as *data distribution*:

- the *null hypothesis* (H_0), which is the statement to be tested. It expresses a condition on a data distribution parameter (like the mean or the variance) or on the data distribution itself (like normal, Poisson, uniform, ...). Equivalently, the null hypothesis consists of a set of possible data distributions, those for which the statement is true. If a unique data distribution is determined, then the null hypothesis is said *simple*, otherwise it is said *composite*.
- the *alternative hypothesis* (H_A), which is the statement considered alternative to the null hypothesis. While it can in principle consist of a unique data distribution, it is typically defined as the negation of the null hypothesis, thus consisting of the infinite set of data distribution complementary to the null hypothesis (this point will be reconsidered in §2.1.4 and §4.3).

Three examples of null hypothesis follow.

- null hypothesis #1: *the mean year salary in the industry sector in Italy is more than 25 k€*. In this case the statement is about a distribution parameter (the mean); equivalently, the (composite) null hypothesis can be described as the (infinite) set of salary distributions whose mean value is more than 25 k€. The alternative hypothesis here is that the mean value is less than or equal to 25 k€.
- null hypothesis #2: *in the population of all college students, the height is normally distributed*. In this case the statement is about a distribution type (normal); equivalently, the (composite) null hypothesis can be described as the (infinite) set of normal height distributions. The alternative hypothesis here is that the height distribution is not normal.
- null hypothesis #3: *given a random generator of numbers in a certain set, numbers are extracted according to a uniform distribution*. In this case the statement is about a precisely defined distribution; equivalently, the (simple) null hypothesis can be described as the (unique) uniform distribution on the given set of possible output numbers. The alternative hypothesis here is that the extraction process distribution is not the uniform one.

Notice that null hypothesis #3 is precisely the main case we are going to consider in this thesis, see Chapter 3 (§3.3.5), Chapter 4 (from §4.4 onwards), Chapter 5 and Chapter 6. Here we also observe that the random variable describing the sample extraction process can be continue (as in null hypothesis #1 and in null hypothesis #2) or discrete (as in null hypothesis #3). In this regard, we anticipate that, while the discrete case is often treated through a continuous approximation, it requires a careful and specific analysis in order to avoid subtle but critical mistakes (see Chapter 3).

Given an observed data sample randomly extracted from the sample space, the purpose of the test is to confirm H_0 or to reject H_0 in favour of H_A, on the basis of the comparison of the observed sample with the expected data based on the (theoretical) model associated with H_0. If data are *consistent*, then H_0 is accepted, otherwise it is rejected in favor of H_A. Here *consistent* means that any difference between observed data and expected data can be attributed to chance[1] (according to some criterion to be defined) and does not reflect a structural difference between the two sets.

From the methodological point of view, the null hypothesis is what is assumed to be true unless the experimental evidence, quantified through the hypothesis test, indicates that the alternative hypothesis is true. Consequently, accepting the null hypothesis means in fact that the tests are not able to give sufficient evidence to support the alternative hypothesis.

2.1.1 Test Statistic

In practice, given a data sample a hypothesis test works as follows. A real value, called *test statistic*, is associated[2] to the given data sample, and then a conclusion is drawn by comparing the test statistic with the pre-defined *acceptance region* (also known as *confidence interval*, made of the values of the test statistic for which H_0 is accepted) and *rejection region* (made of the values of the test statistic for which H_0 is rejected and thus H_A accepted). The process is represented in Figure 2.1. How a given test statistic is associated to the data sample and how the acceptance region and

[1] The term "chance" is commonly used in literature, but to us it appears somewhat inappropriate, as we discuss in §2.1.3.

[2] The kind of association depends on the sample nature. When samples are numeric, the association is typically algorithmic but other types of sample requires of course different definitions, according to the considered setting.

the rejection region are pre-defined constitutes the definition of a specific hypothesis test.

From a more formal point of view, we note that, before being observed, the data sample has to be considered random. Therefore, since the test statistic takes in input a random data sample and provides in output a numeric value, it can be modeled as a random variable, hereinafter referred to as TS, while ts will typically indicate its realization[3].

Hereinafter we assume that the probability distribution associated to the test statistic under the null hypothesis (also referred to as *null distribution* or *sampling distribution*) is known and univocally defined. The above assumption is necessary to build the hypothesis test since knowledge of the null distribution allows to precisely compute the probability that, assuming the null hypothesis to be true, a sample falls in the acceptance region (or in the rejection region) or, alternatively, to choose an acceptance region (or a rejection region) such that the above probability equals some desired value.

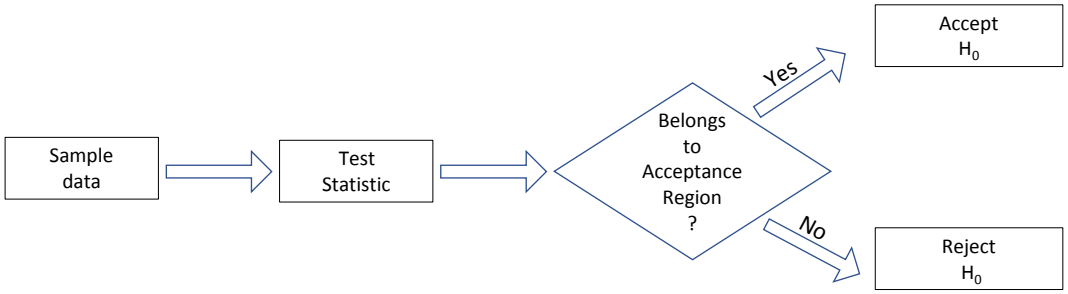


Fig. 2.1 Hypothesis Test, general model

2.1.2 Test conclusions

Given the null hypothesis H_0, the alternative hypothesis H_A, and a data sample, application of the test leads to two possible output conclusions:

- either to accept H_0 (and refuse H_A);
- or to refuse H_0 (and accept H_A).

[3]The realization of a random variable is the value concretely observed in a specific experiment, see also §A.1.

	CONCLUSION	
H_0 REAL VALUE	Accept H_0	Reject H_0
H_0 true	Correct	Type I Error
H_0 false	Type II Error	Correct

Table 2.1 Test configurations

As shown in Table 2.1 we thus have 4 possible configurations, linking the output of the test and the actual value of H_0 (true, false) which, of course, is not known (the goal of a hypothesis test is exactly to infer a conclusion on it).

Two of the above configurations correspond to a correct conclusion of the test

- the null hypothesis is true and the test accepts it;
- the null hypothesis is false and the test rejects it.

The other two configurations correspond instead to an incorrect conclusion of the test

- the null hypothesis is true but the test rejects it (false positive): this type of error is called *Type I*. The probability of encountering a Type I Error is normally indicated with α. When the null hypothesis is rejected we say that the result of the test has *statistical significance* with *significance level* α.
- the null hypothesis is false but the test accepts it (false negative): this type of error is called *Type II*. The probability of encountering a Type II Error is usually indicated with β. Finally $1-\beta$ is called the *test power* and is the probability that the test correctly supports the alternative hypothesis.

Let us now analyze a bit more in depth the meaning of α and β.

2.1.3 Meaning of α

The value α associated to a test represents the probability of false positives, that is, of falling with the test statistic in the rejection region, despite the null hypothesis being true, that is, assuming that the sample is extracted from the sample space according to the data distribution.

Since the null distribution is assumed to be precisely known, in principle it is always possible to know the probability of each test statistic to be observed (according to the underlying null distribution) and thus precisely calculate the value of α associated to the test.

We observe that the meaning of α is often described in literature as the probability to fall in the rejection region *by chance*, but we believe this is a bit misleading description, somehow implying that falling into the acceptance region is normal and correct, while falling into the rejection region is an error, which happens *by chance*. Instead, assuming the null hypothesis is true, we expect that the relative frequency of the observed data samples falling in the rejection region is precisely α, by the very definition of Type I Error. Thus, if we analyse N data samples, extracted according to the data distribution, we expect that about $N\alpha$ of them fall in the rejection region, hence, not *by chance*, but precisely because this is what is expected. Analogously we expect that about $N(1-\alpha)$ times an observed data sample falls in the acceptance region.

In practice, if α is low, then falling in the rejection region, assuming the null hypothesis is true, happens rarely (for example, if we analyse a data sample and α is set to 0.01, we know that we fall in the rejection region on average just once in a hundred). Hence, when it happens, we are quite confident there is some structural reason behind the event, that is, the null hypothesis is probably not true. However we can be unlucky and conclude that null hypothesis is false (Type I Error) when this is not the case. In order to reduce this risk, we can choose a smaller α, e.g. .001, but of course in this way we raise the risk to deem the null hypothesis as true when it is not (Type II Error). The optimal choice for α is left to the analyst, according to the application scenario. The value of α is in principle arbitrarily chosen as a test parameter, however 0.01 and 0.05 are the most used values in literature.

2.1.4 Meaning of β

The value of β, instead, represents the probability of false negatives, that is, of falling with the test statistic in the acceptance region, despite the null hypothesis being false. When the alternative hypothesis is also precisely defined by a unique probability distribution, then it is possible to calculate β (and it can be shown that as α increases β decreases and vice versa).

Much more often, however, the alternative hypothesis is defined simply as the negation of the null hypothesis and, therefore, there are infinitely many possible alternative distributions that describe it. Consequently, computing the value of β is, in general, very hard[4] and, hence, we are normally limited to focusing on the value of α.

2.2 Evaluation methods

As described in §2.1.1, the acceptance region and the rejection region are part of the definition of a hypothesis test. Given the test statistic of an observed data sample, in order to decide if the null hypothesis H_0 has to be accepted or rejected (or, equivalently, to define the acceptance region and the rejection region) two methods are typically used: *the critical value* method (described in §2.2.1) and *the p-value* method (described in §2.2.2). As shown in §2.2.2, the latter can be interpreted as an (equivalent) extension of the former.

2.2.1 Critical Value method

The *critical value method* requires the definition of one or more *critical value(s)*, which act as cut-off point(s) separating the acceptance region and the rejection region. Then, for each data sample, the null hypothesis H_0 is rejected if the test statistic is equal to or more *extreme* (in the direction of the alternative hypothesis, as better precised later) than the critical value; otherwise it is accepted.

Given a data sample, the associated test statistic ts and a critical value CV, three models are normally used to draw a conclusion about the null hypothesis, leading to different meaning of the word *extreme*. For each of them a reference figure is given to better illustrate the model, assuming that a bell-shaped normal curve describes

[4] In order to compute the expected value of β varying across the set of all the possible probability distributions, we should know their probability distribution as well, which is in practice quite unrealistic. However, when this distribution of distributions is known, then computation is in principle feasible. We anticipate that in §4.3 we consider the specific case where the alternative distribution is uniformly taken from the set of all the possible distributions.

the null distribution, that is, the distribution of the values that the test statistic ts can take[5]:

- in the left-tailed model, *extreme* means *too small*: the null hypothesis is rejected if $ts \leq CV$. As an example we can refer to Figure 2.2.
- in the right-tailed model, *extreme* means *too big*: the null hypothesis is rejected if $ts \geq CV$. As an example we can refer to Figure 2.3.
- in the two-tailed model, *extreme* means *too small or too big*: we actually have two critical values CV_L and CV_R, defining a left region and a right region, whose union determines the actual rejection region. The null hypothesis is rejected if $ts \leq CV_L$ or $ts \geq CV_R$. Of course this model can be seen as a combination of the two previous ones. As an example we can refer to Figure 2.4.

In all the three referenced figures, the critical value(s), separating the acceptance region and the rejection region, determines the value of the Type I Error probability, α, defined as the area underlying the Probability Density Function (PDF)[6] of the test statistic values covered by the rejection region. However, what normally happens is that a value is first chosen for α and then the critical value is derived according to the expected (theoretical) probability distribution associated to H_0 and to the value of α.

The choice of the model to be used is in practice is determined by the goal of the statistical test. If we consider again example #1 in §2.1 the null hypothesis is that the year average salary is *more* than 25 k€ and, therefore, the alternative hypothesis is that it is *less* than or *equal* to that amount. The test statistic ts is exactly the year average salary. We then reject the null hypothesis when the test statistic is too small ($ts \leq 25$ k€): we are thus in the left-tailed model.

[5]The assumption of a bell-shaped normal curve (for example purposes) is arbitrary but, in most cases, quite natural. For example it holds in the very frequent case where we randomly extract samples from a given data set and take their means as our test statistic. In this setting, applying the Central Limit Theorem, whatever the data distribution is, it can be proved that as the sample size increases the resulting null distribution approaches a normal distribution. However we anticipate that not all the test statistics can be modeled in this way. As an example, in §6.3.2, we will use a test statistic following the Gumbel distribution, which cannot be approximated with a normal distribution.

[6]The PDF of a *continuous* random variable is the function that describes the likelihood of the outcomes, providing a continuous representation of the probability distribution. See §A.2.3 for a more formal definition.

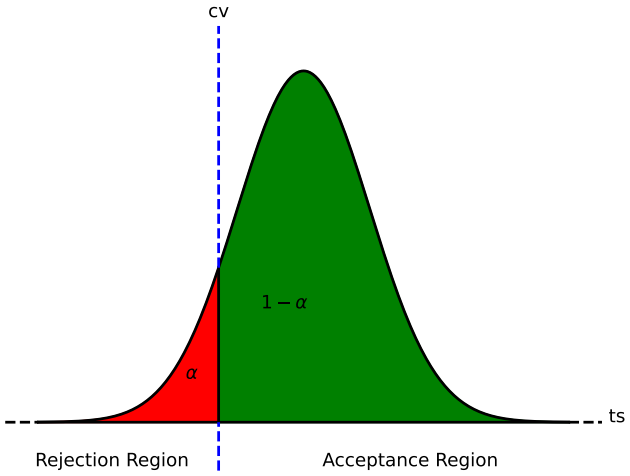


Fig. 2.2 Left-tailed model

Conversely, if our null hypothesis were that the year average salary is *less* than 25 k€, then we would be in the right-tailed model, rejecting the null hypothesis when the test statistic is too big ($ts \geq 25$ k€).

Finally, if the null hypothesis were that the year average salary is *more* than 20 k€ and *less* than 30k€, then we would be in the two-tailed model, rejecting the null hypothesis when the test statistic is too small or too big ($ts \leq 20$ k€ or $ts \geq 30$ k€).

The three mentioned models are commonly used since they are easier to manage and result satisfactory for most situations. However we emphasize that any alternative model is by itself correct, provided it is consistently defined. For example, one may consider a rejection region consisting of an arbitrary number of disjoint sub-intervals of the acceptance region (through the definition of consistently many critical points), obtaining a valid (but scarcely practical) model[7].

2.2.2 P-value method

The *p-value method* relies on the *critical value method*, adding a probabilistic interpretation in order to make the testing process more friendly and the results easier

[7] More considerations on this are proposed in §4.6.3.

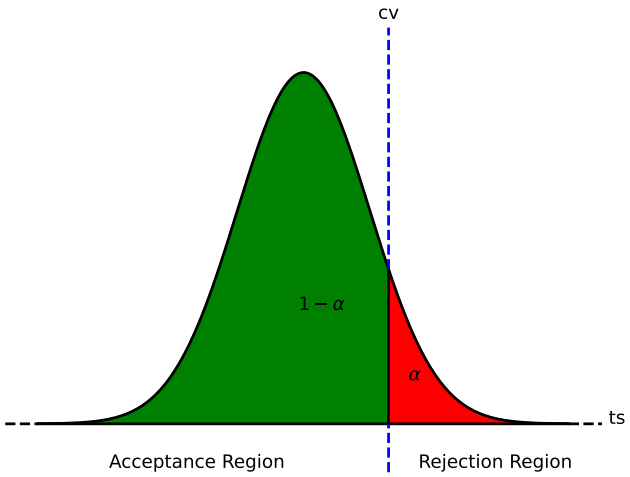


Fig. 2.3 Right-tailed model

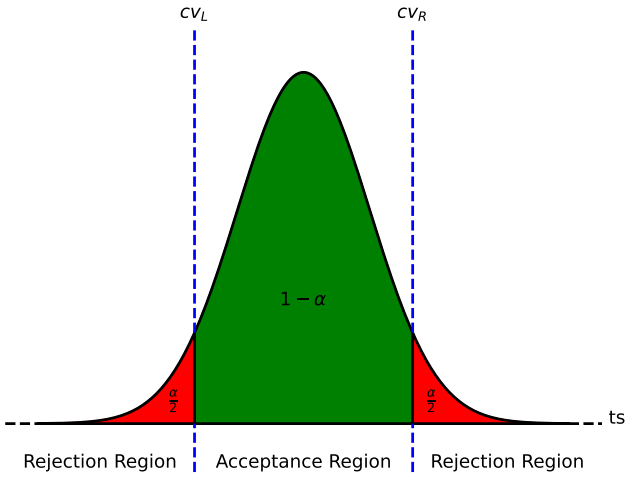


Fig. 2.4 Two-tailed model

to interpret. In the *p-value method* a value pv is associated to each test statistic ts according to Definition 1.

Definition 1. *Given a test statistic value ts, the corresponding p-value* $pv = PV(ts)$ *represents the probability under the null hypothesis of obtaining a test statistic equal to or more* extreme *than ts.*

In other words, if we run an experiment, take the observed data and compute the corresponding test statistic value ts_{obs}, then we know that the *a priori* probability to obtain a value at least as extreme as the one actually observed (ts_{obs}) was exactly $PV(ts_{obs})$, assuming the null hypothesis being true.

We remark that a very common misconception is that the p-value represents the probability that the null hypothesis is correct, which is subtly but clearly different from what stated in Definition 1 (we also observe that the mentioned interpretation cannot be correct since in the p-value computation the null hypothesis is assumed to be certainly -and not probabilistically- true).

It is worth observing that, being a function of a random variable (the test statistic), the p-value is a random variable as well (see [14]), which hereinafter is referred to as PV, whereas by pv we indicate its realization.

Given a test statistic ts, the p-value is calculated using the sampling distribution of the test statistic under the null hypothesis. The exact definition of the p-value $pv = PV(ts)$ depends on the considered model. In the left-tailed model, pv expresses the probability that, under the null hypothesis H_0, the observed test statistic is *equal* to or *smaller* than ts. Equivalently, pv can be defined through the Cumulative Distribution Function (CDF)[8] F_{H_0} associated to the null distribution

$$pv = Pr_{H_0}(t \leq ts) = F_{H_0}(ts) \tag{2.1}$$

where t is taken according to the null distribution. The definition of pv for the right-tailed model is symmetrically given, with pv expressing the probability that, under the null hypothesis, the observed test statistic is *equal* to or *bigger* than ts.

$$pv = Pr_{H_0}(t \geq ts) = 1 - F_{H_0}(ts) \tag{2.2}$$

[8]The CDF of a random variable is the function providing the probability that a random variable is less than or equal to a specified value. See §A.2.3 for a more formal definition.

Model	p-value
left-tail	$pv = Pr_{H_0}(t \leq ts) = F_{H_0}(ts)$
right-tail	$pv = Pr_{H_0}(t \geq ts) = 1 - F_{H_0}(ts)$
two-tailed	$pv = 2Pr_{H_0}(t \geq \|ts\|) = 2(1 - F_{H_0}(\|ts\|))$

Table 2.2 P-value and test statistic relation

Finally the two-tailed model is a sort of combination of both. Assuming for simplicity that the null distribution is symmetric about the origin[9], pv can be defined as:

$$pv = Pr_{H_0}(t \leq -|ts|) + Pr_{H_0}(t \geq |ts|) = 2(1 - F_{H_0}(|ts|)) \tag{2.3}$$

with pv expressing the probability that, under the null hypothesis, the observed test statistic is *equal* or *smaller* than ts if $ts < 0$ or *equal* or *bigger* than ts if $ts \geq 0$. The three cases are summarised in Table 2.2.

In each case, in order to decide if H_0 has to be accepted or rejected, pv is then compared to the probability α as defined in §2.1.3, that is, the probability that under the null hypothesis a sample falls in the rejection region. If

$$pv \leq \alpha$$

then the null hypothesis is rejected, otherwise it is accepted.

Notice that in literature the criterion that determines whether the test statistic associated to a given sample belongs to the rejection region is expressed both as

$$pv < \alpha$$

and

$$pv \leq \alpha \tag{2.4}$$

We observe that the two formulations are equivalent when the underlying null distribution is continuous because in this case the probability of the test statistic taking on any specific value (including α) is zero.

[9]The assumption that the distribution is symmetric is quite common and reasonable, see for example Footnote 5 at page 17. Moreover, simple transformation can often be applied to translate it around the origin (think for example of the Normal Distribution Standardization, transforming an arbitrary Normal Distribution in the Standard one $N(0,1)$, see §A.4.3.2).

However, if the distribution is discrete, then the equality shown in Equation (2.4) should be considered, because by definition α is precisely the probability to fall in the rejection region under the null hypothesis.

Observation 1. *According to the above description, given a sample, the p-value method can be seen (1) as producing a hard decision about the null hypothesis (accept, reject); or (2) as providing a p-value that is then compared with an external threshold given by the Type I Error α. Hereinafter we will adopt the second interpretation (2), because more flexible (as later discussed in §2.2.3.2).*

A methodological remark can be made here. As well discussed in [15], a testing procedure requires that a data-independent decision rule is set in advance, before the data analysis. Thus, in our case, this means that first we set the (data-independent) Type I Error probability α, then we observe data samples and compute the corresponding (data-dependent) p-values, thus taking the final decision (accept/reject the null hypothesis) by comparing the obtained p-value with the pre-defined value of α. While this may look trivial, misinterpretation of the exact relation between Type I Error probability and p-values is quite common. In [15] interesting considerations at this regard are proposed.

2.2.3 Comparison of the two methods

We can now compare the two methods. First of all, we observe that the two methods have different domains. The critical value method works on the set of values that the test statistic can assume (for example the year salary, as in the case #1 in §2.1, or the stature of a group of people) and the acceptance region and the rejection region are defined by the critical value(s). The p-value method, on the contrary, operates in the probability realm, therefore in the set of values [0,1], and the acceptance region and the rejection region are defined by the parameter α.

The two concepts are however closely related, since p-values are obtained simply applying to the test statistic values the associated CDF F_{H_0}, as per Table 2.2, while α and the critical value(s) CV are linked by the relation shown in Table 2.3.

Figures 2.5, 2.6 and 2.7 represent the relation between the two methods in a simple example in the left-tailed model. More precisely, Figure 2.5 contains the sample space (the lilac cloud, with a subset where each gray circle is a sample) and

Model	α
left-tailed	$\alpha = F_{H_0}(CV)$
right-tailed	$\alpha = 1 - F_{H_0}(CV)$
two-tailed	$\alpha = F_{H_0}(CV_L) + 1 - F_{H_0}(CV_R)$

Table 2.3 Computation of α from the critical value(s)

the test statistic range on the real axis, with the critical point CV (the small gray vertical segment) separating the rejection region (red, on the left) and the acceptance region (green, on the right).

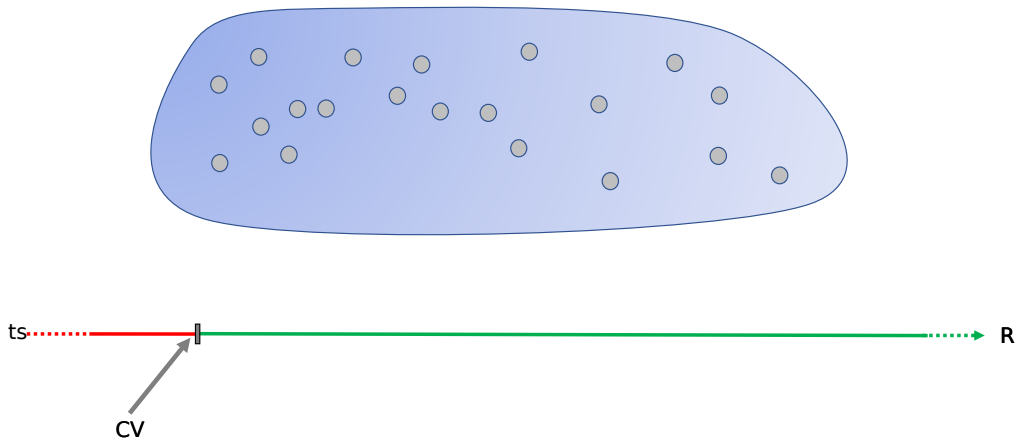


Fig. 2.5 Sample space and test statistic axis in the left-tailed model

Figure 2.6 in addition maps each sample on a test statistic value on the real axis. Red samples are those mapped on a a red star on the real axis, corresponding to a test statistic value less than or equal to the critical value, and thus fall in the rejection region. Conversely, green samples are those mapped on a a green star on the real axis, corresponding to a test statistic bigger than the critical value, and thus fall in the acceptance region.

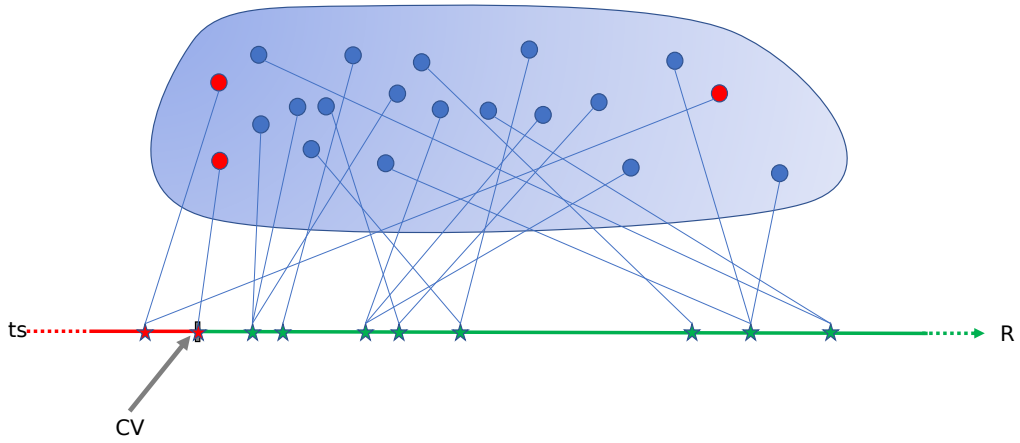


Fig. 2.6 Samples mapped on the test statistic axis in the left-tailed model

Finally Figure 2.7 maps each test statistic value (red or green star) onto a p-value (a little circle with the same color of the star), where the critical value is mapped onto the Type I Error α, separating rejection and acceptance regions. We thus have a mapping from the possibly infinite test statistic domain on the $[0, 1]$ probability range.

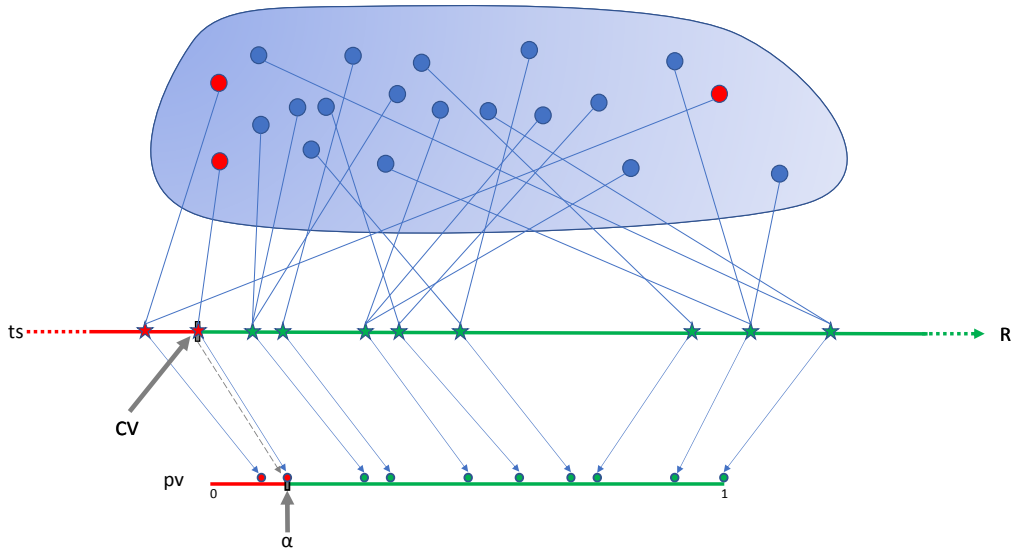


Fig. 2.7 Test statistic values mapped on the p-values in the left-tailed model

Figures 2.8 shows the relation between the critical value method and the p-value method in the right-tailed model. A proper combination of Figures 2.7 and 2.8 would provide a representation of the two-tailed model, here skipped for simplicity.

Model	critical value method	p-value method
left-tailed	$ts \leq CV$	$pv \leq \alpha$
right-tailed	$ts \geq CV$	
two-tailed	$ts \leq CV_L$ or $ts \geq CV_R$	

Table 2.4 Rejection region criterion

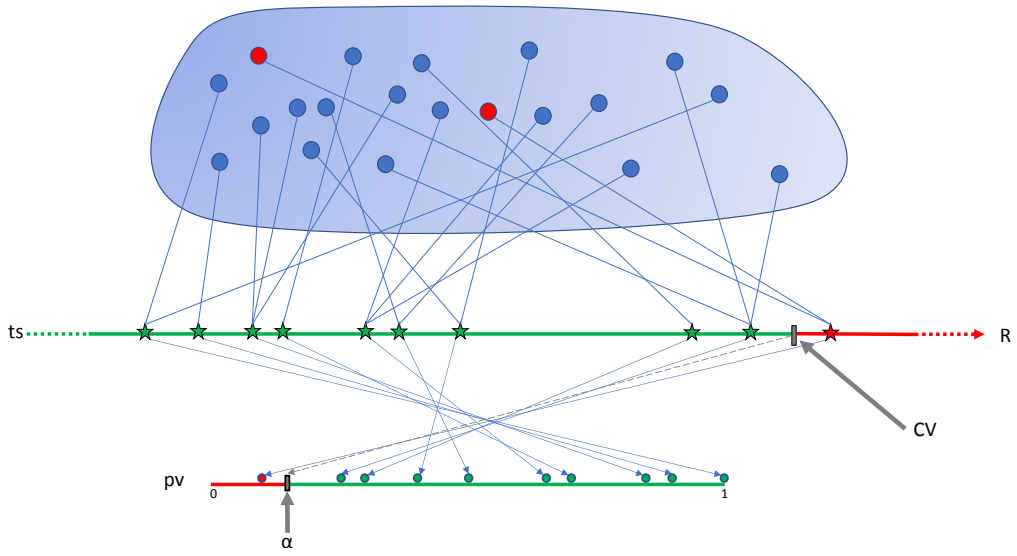


Fig. 2.8 Test statistic values mapped on the p-values in the right-tailed model

2.2.3.1 Methods equivalence

Although the *p-value* method may look more general than the *critical value* method, in fact they are equivalent: for any given critical value CV and test statistic ts we can always make use of the CDF F_{H_0} associated to the probability distribution of the test statistic (see Equations (2.1), (2.2) and (2.3)) and indifferently apply either method.

In fact, if we consider the left-tailed model, we set

$$pv = F_{H_0}(ts), \alpha = F_{H_0}(CV) \tag{2.5}$$

and we equivalently, reject the null hypothesis both when $ts \leq CV$ (*critical value* method) and when $pv \leq \alpha$ (*p-value* method). The same equivalence clearly holds for the right-tailed model and the two-tailed model, with the obvious modifications shown in Table 2.4.

The equivalence between the two methods above described can also be seen from an information theory point of view, observing that the random variable $PV = PV(TS)$, despite being typically interpreted as a probability, can also be seen as a bijection between the test statistic range and a bounded interval which, chosen by convenience equal to the probability range $[0,1]$, allows the probabilistic interpretation (see for example [15] for a more formal analysis). Being a bijection, the p-value cannot add or remove any information already provided by the corresponding test statistic, thus explaining, in information theory terms, why the two methods are indeed equivalent.

2.2.3.2 Practical considerations

Although the two methods are in principle equivalent when α is fixed (as shown above), the *p-value method* can be preferred in practice because it is more versatile. Thanks to the wide availability of software tools implementing the most common probability distributions underlying the null hypothesis, and, therefore, able to derive the corresponding CDF F_{H_0} (see Equation (2.5)), in many cases we can easily compute the p-value *pv* for a given data sample. While in the critical value method we can just compare the test statistic with one single pre-defined critical value (or pair of values), in the p-value method we can compare the resulting *pv* with any value of α chosen according to the test needs (in this sense we conveniently consider α as an external parameter of the p-value method, as anticipated in Observation 1). However, even if the p-value provided by the p-value method apparently gives the analyst a richer information than the binary result of the critical value method, it is worth realizing that the advantage is practical and not conceptual. Indeed, to decide when the null hypothesis should be accepted or rejected, the analyst must ultimately set an explicit limit value (α), which in the critical value method is simply hidden in the critical value(s). In this regard, see also the methodological remark at the end of §2.2.2.

A second reason to prefer the p-value method is that the meaning of the test statistic, as provided by the critical value method, depends on the specific application and its value varies in a specific scenario-dependent range. As a consequence the test statistic value cannot be directly used for comparison. On the contrary the p-value

provides a probability, which is very easy to interpret and has the same meaning for any test and thus allow to directly compare values[10].

[10]As a toy example, given an 9-years old female, 128 cm tall and 27 kg heavy, with the critical value method we cannot say if she is, in a sense, taller or heavier (according to the respective known distributions). However with the p-value method we can rely on children growth charts, which associate a p-value to each height and another p-value to each weight (in this interpretation p-values are normally referred to as *percentiles*). Assuming the p-values provided by the charts are $p_H = .32$ for 25 kg and $p_W = .14$ for 125 cm, since $p_W > p_H$ we can conclude that the girl is heavier than she is tall.

Chapter 3

P-values analysis

As discussed in Chapter 2, given a hypothesis test, a real-valued test statistic is associated to each sample taken from the sample space. The test statistic can be interpreted as a random variable and, as such, has an associated probability distribution, said null distribution. Furthermore, if the test is implemented according to the p-value method (as commonly happens in practice and as assumed hereinafter), an additional (probability) p-value is derived from the test statistic and represents the output of the test[1]. The p-value can be interpreted in turn as a random variable with an associated probability distribution. The object of the chapter is the study of the probability distribution of the p-value and other related properties.

In particular, in §3.1 we consider the case of a continuous null distribution observing that the resulting p-value distribution is uniform on [0,1] independently of the underlying null distribution. However, when we move to the discrete null distribution case, things unfortunately turn out to be much less regular. Most of the chapter (from §3.2 to the end) is, therefore, devoted to the discrete case.

Hence, in §3.2 we introduce the discrete setting and in particular the finite case. Then we propose the concept of *p-tuple*, that is, the list of all the observable p-values for a given test, which turns out to be a basic tool for the subsequent analysis. Finally, we study the probability distribution functions of the p-value random variable.

[1] As shown in §2.2.2, the resulting p-value is then compared with a pre-defined threshold in order to take a hard decision about accepting or rejecting the null hypothesis on the basis of the observed sample.

Then, in §3.3, we analyze the p-tuples in relation to the data distribution of the sample space. In particular, after providing the general model and some useful related concepts, we characterize the form the p-tuples can assume, according to the underlying data distribution. Three cases are examined, each with increasingly specific definitions: when no constraint on the data distribution is given, when the data distribution is arbitrary but fixed, and ultimately, when the data distribution is uniform. The latter case is particularly significant to us as it encompasses random number generators with a uniform probability distribution.

Finally, the following Section §3.4 is devoted to the case of uniform distribution of the p-values. In particular its application as a basis to build a meta-test for the validation of the null hypothesis is considered.

3.1 The continuous case

If the null distribution is continuous, by definition of p-value (as the probability under the null hypothesis that the observed test statistic is equal to or more *extreme* than a given value (see Definition 1)), a very important condition follows, mentioned for example in [16] and precised by the following Theorem 1.

Theorem 1. *If the null hypothesis is satisfied and the null distribution is continuous, then the p-value is uniformly distributed.*

Proof. For simplicity let us consider the *left-tailed* model[2]. The p-value PV can be expressed as a function of the test statistic TS

$$PV = F_{H_0}(TS)$$
$$pv = F_{H_0}(ts)$$

where F_{H_0} is the CDF of the random variable TS under the null hypothesis H_0. Let us now compute the CDF F'_{H_0} of the random variable PV. Assuming that F_{H_0} is

[2]Proofs for the *right-tailed* and *two-tailed* models follow the same path.

invertible[3], for any pv we have

$$\begin{gathered} F'_{H_0}(pv) = \\ Pr(PV < pv) = \\ Pr(F_{H_0}(TS) < pv) = \\ Pr(TS < F^{-1}_{H_0}(pv)) = \\ F_{H_0}(F^{-1}_{H_0}(pv)) = \\ pv \end{gathered} \tag{3.1}$$

From Equation (3.1) we then have $F'_{H_0}(pv) = pv$ which is equivalent to saying that PV is uniformly distributed and, thus, proves the theorem. □

The condition of uniform distribution of the p-value can be equivalently expressed as

$$Pr(PV \leq a|H_0) = a, \forall a \in [0,1] \tag{3.2}$$

We observe that the nice property proved in Theorem 1 about the CDF of the p-value follows from the fact that the p-value is itself a CDF (of the test statistic). Remarkably, the uniformity proved in Theorem 1 means that, in the considered setting of continuous null distribution, the p-value distribution is independent of the underlying null distribution. While this independence may appear surprising, it can be explained observing that the null distribution is actually encompassed in the definition of the p-value[4]. In Figures 3.1 and 3.2 the PDF and the CDF of the p-value, respectively, for an arbitrary continuous null distribution are represented.

[3]The invertibility of F_{H_0} in Equation (3.1) can be assumed with little loss of generality, since it holds in most frequently used models, like for example when the null distribution follows a normal, a uniform, a chi-squared or other common distributions (see §A.4).

[4]In fact the null distribution encompasses in turn both the extraction process of the samples (described by the data distribution) and the test statistic function mapping a sample on a test statistic. The same test statistic with a different data distribution would in general result in a different probability distribution of the p-value. Likewise, given the same data distribution with a different definition of the test statistic, we likely obtain a different distribution of the p-value.

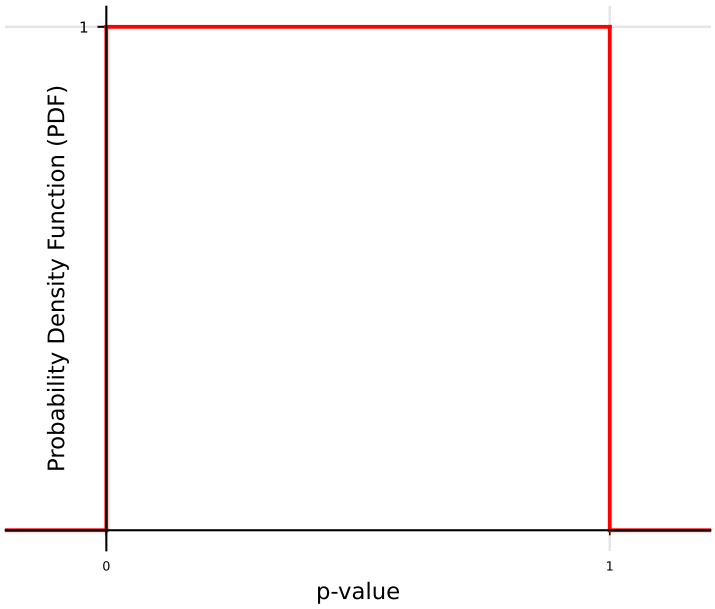


Fig. 3.1 P-value Probability Density Function (continuous case)

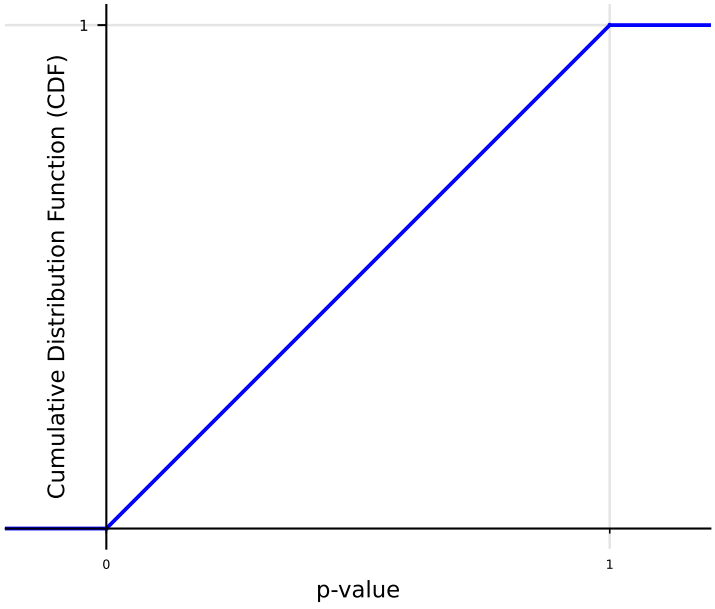


Fig. 3.2 P-value Cumulative Distribution Function (continuous case)

3.2 The discrete case

In practice however we often work with discrete distributions which, for ease of calculation, are replaced by their continuous approximations. While this works very well in most cases, it sometimes leads to assumptions and results that are incorrect due to the specific discrete nature of the setting.

For example in the statistical test suite proposed by NIST [17] to analyse binary sequences produced by random generators (see §5.2), one of the decision criteria is based on the assumption that the p-values are uniformly distributed, as per Theorem 1. However, as we will see in §3.4, this is conceptually incorrect because of the discrete nature of binary sequences and turns out to be practically imprecise even for some tests of the suite, as shown in §5.2.2.2.

For this reason hereinafter we elaborate on the specific setting of a discrete null distribution, which implies that the distribution of the p-values is necessarily discrete as well. As a consequence, Equation (3.2) cannot hold, because infinitely many values $a \in [0,1]$ are not observable p-values (that is, it is impossible that they appear) and the equality cannot be satisfied.

Because of the very definition of p-value, however, as shown in [18], the p-value random variable PV still partially satisfies Equation (3.2) in the (weaker) sense that, assuming the null hypothesis true, for any given test,

$$Pr(PV \leq \omega | H_0) = \omega, \forall \omega \in \Omega \tag{3.3}$$

where Ω is the set of observable p-values (or, equivalently, Ω is the support of PV) for the considered test, as we introduce in the following definition

Definition 2. *For a given test, let Ω be the set of p-values ω for which there exists at least one test statistic value ts such that $PV(ts) = \omega$.*

Equation (3.3) defines a sort of uniformity, which will be reconsidered in §3.4. We anticipate however that it does not imply *discrete uniformity*, that is the property that all the elements of Ω have the same probability: $Pr(PV = \omega_1) = Pr(PV = \omega_2), \forall \omega_1, \omega_2 \in \Omega$ (see later Definition 11).

We also note that the set Ω of Definition 2 strictly depends on the specific test we are considering and hence Equation (3.3) depends on the considered test as well,

while Equation (3.2) holds for an arbitrary test in the continuous setting, as earlier observed.

3.2.1 P-tuples

Let us now consider in particular the discrete and finite case, where the number of observable test statistic values is finite and therefore the number of distinct p-values is finite as well[5]. We can then write

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_{N_\Omega}\}$$

Assuming, without loss of generality, that

$$\omega_i < \omega_{i+1}, \forall i \in [1, N_\Omega - 1]$$

then Ω can also be seen as a tuple (that is, an ordered list) and more consistently indicated as *p-tuple* and referred to with the notation

$$\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$$

3.2.2 P-value probability distribution functions

Being in the finite case, if we want to analyse the distribution of the p-values we do not deal with the PDF but instead with the Probability Mass Function (PMF)[6] of the p-value PV, hereinafter indicated by the notation $f_{PV}(\omega) = Pr(PV = \omega)$. An interesting thing here is that the p-tuple Ω and the PMF $f_{PV}(\omega)$ completely define each other, as shown below.

Since $\omega_i < \omega_{i+1}, \forall i \in [1, N_\Omega - 1]$, Equation (3.3) can be re-written as

$$\omega_i = \sum_{j=1}^{i} f_{PV}(\omega_j) \tag{3.4}$$

[5]This is the setting of our main interest, since it captures also the case of random generators producing bit sequences of a given finite length.

[6]The PMF of a *discrete* random variable is the function that assigns the probability to be observed to each possible value of the variable. See §A.2.2 for a more formal definition.

or, equivalently,

$$\begin{aligned} \omega_1 &= f_{PV}(\omega_1) \\ \omega_i &= f_{PV}(\omega_i) + \omega_{i-1}, \forall i \in [2, N_\Omega] \end{aligned} \tag{3.5}$$

and reciprocally,

$$\begin{aligned} f_{PV}(\omega_1) &= \omega_1 \\ f_{PV}(\omega_i) &= \omega_i - \omega_{i-1}, \forall i \in [2, N_\Omega] \end{aligned} \tag{3.6}$$

In addition, as for any probability distribution, given the PMF f_{PV}, the CDF F_{PV} is completely determined (Equation (3.7)) and vice versa (Equation (3.8)).

$$F_{PV}(pv) = \sum_{\omega_i \leq pv} f_{PV}(\omega_i) \tag{3.7}$$

$$\begin{aligned} f_{PV}(\omega_i) &= F_{PV}(\omega_i) - F_{PV}(\omega_{i-1}), \forall i \in [2, N_\Omega] \\ f_{PV}(\omega_1) &= F_{PV}(\omega_1) \end{aligned} \tag{3.8}$$

As an example, let us now consider Figure 3.3, representing again the setting considered in Figure 2.7, extended with an arbitrarily chosen p-tuple Ω of observable associated p-values[7]

[7]For example purposes, the p-values listed in Equation (3.9) are determined assuming uniform sampling from the lilac cloud (in this setting, the probability to observe any given test statistic value ts is proportional to the number of samples $u \in U$ mapped onto ts). Any other arbitrary set of distinct p-values would work as well (with the only constraint that the set represents a *valid* tuple, according to Definition 8 given later in §3.3.4.2).

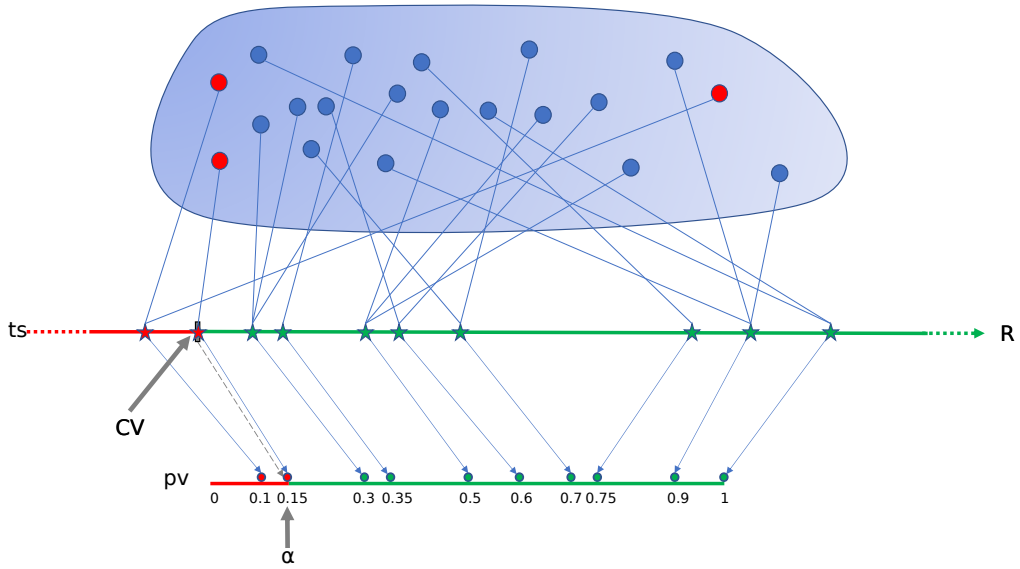


Fig. 3.3 Left-tailed model with p-values

$$\Omega = \{0.1, 0.15, 0.3, 0.35, 0.5, 0.6, 0.7, 0.75, 0.9, 1\} \tag{3.9}$$

As shown by Equation (3.6), the PMF of the p-values is completely determined by the above tuple Ω (see Equation (3.9)) and is represented in Figure 3.4

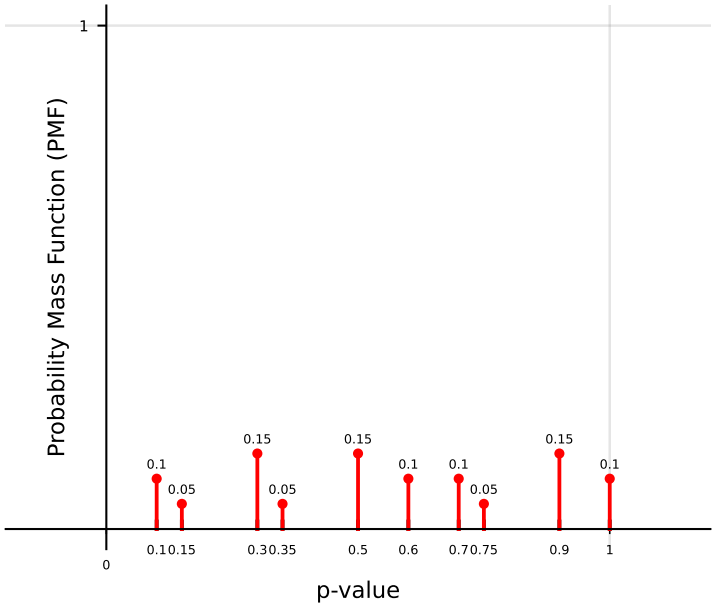


Fig. 3.4 P-value Probability Mass Function (discrete case, example)

which gives the probability of PV being equal to any value of Ω:

$$f_{PV}(0.1) = 0.1, f_{PV}(0.15) = 0.05, f_{PV}(0.3) = 0.15, \ldots, f_{PV}(0.9) = 0.15, f_{PV}(1) = 0.1$$

The resulting staircase CDF is represented in Figure 3.5, according to Equation (3.7).

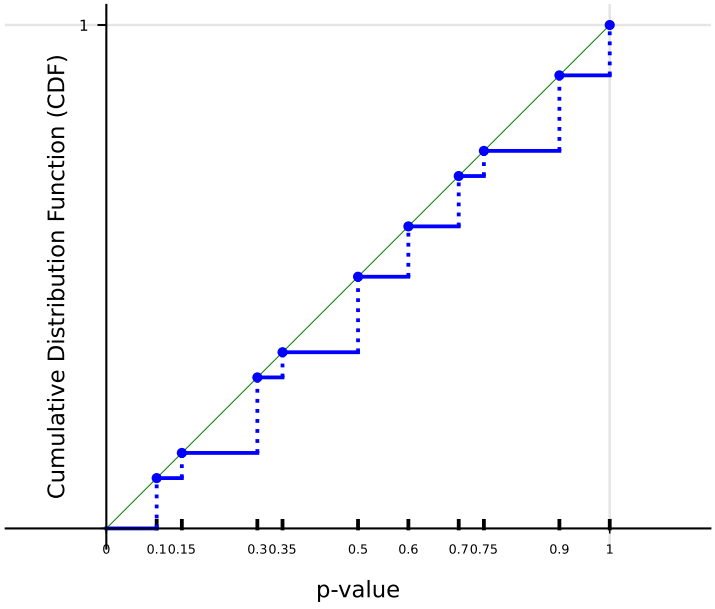


Fig. 3.5 P-value Cumulative Distribution Function (discrete case, example)

Hence, we can make the following observation

Observation 2. *Given a test and the associated p-value variable PV, then PV can be equivalently described by its PMF f_{PV}, its p-tuple Ω (see Equations* (3.4)*,* (3.5) *and* (3.6)*) or its CDF F_{PV} (see Equations* (3.7) *and* (3.8)*).*

In order to better describe the p-values behaviour, we can now make some simple but useful considerations about their distribution.

3.2.2.1 Extreme p-values

First, from Equation (3.4), setting $i = N_\Omega$ and taking into account that, being f_{PV} a PMF,

$$\sum_{i=1}^{N_\Omega} f_{PV}(\omega_i) = 1$$

we derive the following

Observation 3. *The biggest value of* Ω *is always equal to* 1 *(that is,* $\omega_{N_\Omega} = 1$*).*

Equivalently, we note that, being in the finite case, we necessarily always have a least *extreme* value for the test statistic, where the meaning of *extreme* depends on the specified model (see §2.2). By Definition 1 of p-value, that extreme test statistic value is mapped on a p-value equal to 1, because any other observable test statistic is certainly more extreme. Thus in a finite distribution we always have $max(\Omega) = 1$, as represented on the right edge of the p-value axis in Figures 2.7 and 2.8. For example, if we refer to the left-tailed model (Figure 2.7), the least extreme test statistic value is the one on the right on the ts axis and the corresponding p-value is indeed 1 on the pv segment.

On the other hand, denoted by ω_1 the minimum value in Ω, we observe that by Definition 2 each ω_i, thus including ω_1, has at least one counter-image. Therefore we have $f_{PV}(\omega_1) > 0$, but also $\omega_1 = f_{PV}(\omega_1)$ by eq. (3.6) and thus $\omega_1 > 0$. Hence, we can make the following remark.

Observation 4. *The smallest value of* Ω *is always strictly positive (that is,* $\omega_1 > 0$*).*

3.2.2.2 P-values distance

Furthermore, we consider the spacing among observable p-values.

Observation 5. *The distance among the p-values (in the (0,1] probability range) does not depend on the distance of the corresponding test statistic values (in* $\mathbb{R}$*), rather on the probabilities associated to them (that is, the cumulative probabilities of extraction of the samples mapped on them and thus the PMF values* f_{PV}*)*[8].

In particular, applying Equation (3.5) we see that the distance between two consecutive p-values is equal to the PMF of the bigger one:

$$\omega_i - \omega_{i-1} = f_{PV}(\omega_i), 1 < i \leq N_\Omega$$

[8]For example, if all the samples are extracted with the same probability from the sample space, as in Figure 3.3, then the probability of a given test statistic is proportional to the cardinality of the set of samples mapped on it.

and in general the distance between two p-values is given by the sum of the PMF of the bigger one and of the intermediate p-values

$$\omega_i - \omega_j = \sum_{k=j+1}^{i} f_{PV}(\omega_k), 1 \leq j < i \leq N_\Omega$$

3.2.2.3 Estimation of the Type I Error probability α

Here we focus for simplicity on the left-tailed model (but similar considerations hold for the right-tailed and the two-tailed model), for which a certain probability value μ acts as separator between the rejection region $RR = [0, \mu]$ and the acceptance region $AR = (\mu, 1]$ on the $[0, 1]$ probability range (see §2.2.2). Recalling that the Type I Error probability α represents the probability $Pr(PV \leq \mu)$ that (the test statistic associated to) a sample, randomly taken from the sample space according to the underlying data distribution, falls in the rejection region, we first note that $\alpha = \alpha_\mu$, that is, α is determined by μ (for a given test).

Then, in the continuous setting, it follows by Equation (3.2) that

$$\alpha_\mu = Pr(PV \leq \mu) = \mu$$

that is, for any μ,

$$\alpha_\mu = \mu$$

However, in the discrete case we know that Equation (3.2) does not hold in general and, in fact, it holds only for the elements of the Ω set, made of the observable p-values, as by Equation (3.3). In particular, for any $\mu \in [0, 1]$, the staircase CDF (see Figure 3.5) shows that

$$Pr(PV \leq \mu) = \begin{cases} 0 \text{ if } \mu < \min \Omega \\ \min\{\omega \in \Omega \text{ such that } \omega \leq \mu\} \text{ otherwise} \end{cases}$$

and, therefore,

$$\alpha_\mu = Pr(PV \leq \mu) \begin{cases} = \mu \text{ if } \mu \in \Omega \\ < \mu \text{ if } \mu \notin \Omega \end{cases} \tag{3.10}$$

Hence, we can make the following

Observation 6. *In the discrete case, the Type I Error probability (α) is equal to the probability separator value between the acceptance region and the rejection region (μ) only if the separator is taken from the set (Ω) of the possible p-values. Otherwise it is smaller.*

In concrete hypothesis testing procedures, the Type I Error probability is typically considered an external parameter (see also Observation 1) and, as such, is set by the analyst. However, what the analyst typically does in practice is to set the μ value, assuming that $\alpha_\mu = \mu$. While this is correct in the continuous case, Observation 6 says that in the discrete case, instead, we always have $\alpha < \mu$ for any μ, but for a countable number of values (those for which $\mu \in \Omega$) where equality holds.

Unfortunately, we also remark that, in general, it is not easy to fully determine the elements of Ω[9]. Hence, when we choose a μ value, we are not always able to precisely compute α_μ. As an extreme case, Equation (3.10) tells that if the separator is (unawarely) chosen smaller than the smallest value of Ω (that is, $\mu < \omega_1$), then the test statistic value can never fall in the rejection region (differently said, it is not just a matter of observing enough samples in order to find one rejection).

Equation (3.10) is also important in correctly building a validation methodology for a given test, that is, fixing a separator value μ, taking a sample space and checking that the number of samples falling in the rejection region tends to α_μ as the sample space size increases. Assuming that $\alpha_\mu = \mu$ can lead to incorrect results.

In view of the preceding considerations, we introduce the following

Definition 3. *A μ value is said* admissible *if it belongs to the set (Ω) of observable p-values.*

From Equation (3.10) we derive that $\alpha_\mu = \mu$ if and only if μ is admissible.

3.3 P-tuples characterization

A deep understanding of the p-value is very important, since the ultimate output of a test is precisely a p-value (which is then compared to a predefined threshold to

[9]Of course, if N_U is small, we can exhaustively compute all the observable p-values. In general, however, it might be prohibitive to analytically determine the whole set Ω.

take the final hard decision, see §2.2.2). As observed in §3.2, the p-value random variable associated to a given test can be equivalently described by its PMF or its CDF but also through the associated p-tuple, formed by the list of the observable p-values produced by the test (for the definition of p-tuple see §3.2.1). Hence, in this §3.3 we study some properties of the p-tuples, giving a characterization under different assumptions[10].

The section is organized as follows. In §3.3.1 we analyse the general model of a hypothesis test, emphasizing the relation among its elements: the sample space, its probability distribution, the test statistic function and the p-value variable. Then in §3.3.2 the probabilities associated to the observable p-values are analyzed and the concept of *ordered set partition* is introduced. In §3.3.3, the definition of *valid* tuple is proposed, allowing to fully characterize the set of the observable p-tuples when no constraint is given on the data distribution. Then, in §3.3.4, we consider the more practical setting with an arbitrary but fixed data distribution: we first introduce the concept of *equivalent tests*; then we give the definition of *F-valid* tuple which, combined with the notion of ordered set partition, allows to characterize the set of observable p-tuples with a given (arbitrary but fixed) data distribution; subsequently we prove a bijection between the set of the ordered set partitions and the set of non-equivalent tests which can be defined under the given data distribution[11]; finally we count the number of non-equivalent tests and derive a necessary and sufficient condition on the data distribution in order to have a distinct p-tuple for each non-equivalent test. Then, in §3.3.5, we consider the case of uniform data distribution, introducing the concept of *U-valid* tuple and determining a characterization of the set of the observable p-tuples in the given setting. Finally, in §3.3.6 a summary of the key points developed in the whole §3.3 is presented.

[10]Hereinafter we focus on the *single* null hypothesis setting, which means that a unique data distribution (that is, the underlying distribution of the sample extraction process) is associated to the null hypothesis (see §2.1), thus allowing to use interchangeably the two concepts of null hypothesis and data distribution. Note that a remarkable example of this setting is the null hypothesis of most interest for us, i.e. when the samples are uniformly extracted (as in the case of the random generators model we are mainly addressing in this dissertation).

[11]We stress that the data distribution is part of the definition of the test, since it impacts on the p-value definition.

3.3.1 Hypothesis test general model

Here we summarize the general model of a hypothesis test when the sample space is finite, as described in §3.2 and shown for example in Figure 3.3. The test can be seen as a 3-steps process:

- **sample extraction**. Given a finite sample space $U = \{u_1, u_2, \ldots, u_{N_U}\}$, each sample $u = u_i$ is extracted with probability $f_V(u_i)$, where V is the random variable modeling the extraction process from U according to the data distribution and f_V is its PMF. In the following the data distribution will thus be described by f_V;

- **test statistic**. The extracted sample u is then mapped by the test statistic function (TS) on a test statistic value $TS(u)$. If we indicate with $T = \{\tau_1, \tau_2, \ldots, \tau_{N_T}\}$ the set of the test statistic values obtained as the sample vary in U, we have $N_T \leq N_U$ since in general more samples can be mapped on the same test statistic value[12]. Without loss of generality we can assume that

$$\tau_i < \tau_{i+1}, \forall i \in [1, N_T - 1] \tag{3.11}$$

  and T can thus also be seen as a tuple (that is, an ordered list) and more consistently referred to with the notation

$$T = (\tau_1, \tau_2, \ldots, \tau_{N_T})$$

- **p-value**. In the next step for each i the test statistic value τ_i is mapped by the p-value function (PV) into a p-value φ_i,

$$\varphi_i = PV(\tau_i) = Pr(TS(u) \leq \tau_i) = \sum_{h=1}^{i} Pr(TS(u) = \tau_h) \tag{3.12}$$

  thus defining the set Φ of the observable p-values

$$\Phi = \{\varphi_1, \varphi_2, \ldots, \varphi_{N_T}\}$$

[12] For example, if the space U is made of all the binary sequences of a given length and the test statistic function counts the number of 1s in a given sequence, then we clearly have multiple sequences colliding on the same test statistic.

whose cardinality is N_T as well because of the bijection between T and Φ[13].

Figure 3.6 exemplifies the process, with $N_U = 12$ and $N_T = 6$. In the figure we also introduce the subsets U_i of the sample space U. Their meaning will be detailed in §3.3.2, here we anticipate that samples of U are grouped in subsets U_i, according to the test statistic τ_i on which they are mapped.

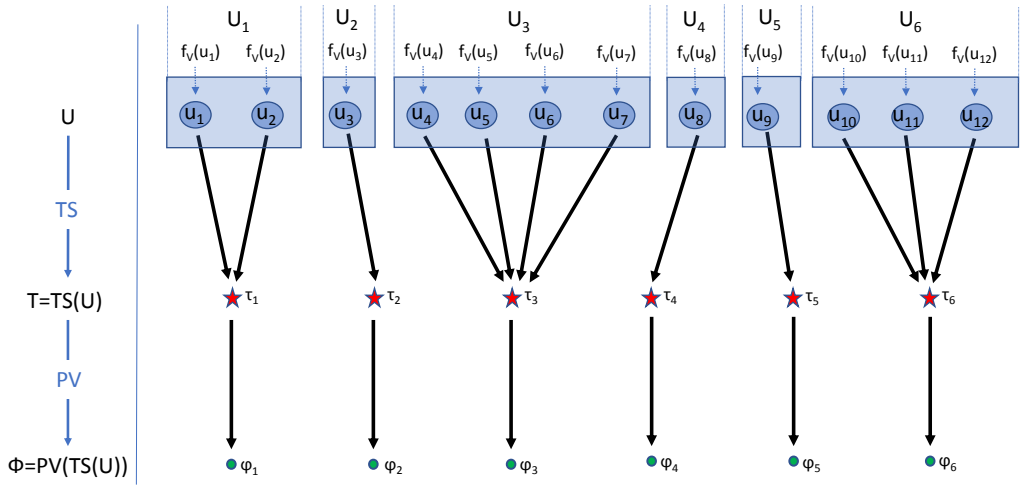


Fig. 3.6 From sample space U to p-values

Relying on the above description, we can make the following observations.

Observation 7. *A hypothesis test is completely defined by three elements:*

- *the sample space U;*
- *its data distribution f_V;*
- *the test statistic function TS, mapping samples $u \in U$ to real numbers.*

The fourth element mentioned in the above description, that is, the PV function mapping the test statistic values on probability values, is not listed in Observation 7

[13]The bijective relation was already anticipated in §2.2.3.1 for the continuous case. In the discrete case we observe that the mapping from T to Φ is a surjection by construction, while the injection can be proved as follows: given τ_i and τ_j, with $i < j$ and therefore $\tau_i < \tau_j$, we have $\varphi_i = Pr(TS(u) \leq \tau_i) = \sum_{h=1}^{i} Pr(TS(u) = \tau_h) < \sum_{h=1}^{j} Pr(TS(u) = \tau_h) = Pr(TS(u) \leq \tau_j) = \varphi_j$ and hence $\varphi_i < \varphi_j$.

because it is completely determined by the others, as already shown in §2.2.2 (see in particular Table 2.2).

We note that Observation 7 is consistent with the claim expressed in §2.2.3.1 that the critical value method and the p-value method are equivalent, since the introduction of the p-value does not add (or remove) any information but it simply reshapes the existing one in a more homogeneous and easy-to-use form.

Finally, on the basis of Observation 7, we give the following definition.

Definition 4. *A triple made of a sample space U, a function f_V and a function TS, is said* legitimate *if f_V represents a data distribution of U and TS is a real-valued function defined on U. Given a legitimate triple, a test can be associated to the triple according to the model given in §3.3.1. For the sake of simplicity, we will also refer to the resulting test as* legitimate*, meaning that it is consistently defined.*

3.3.2 P-values probability

Examining the probability of a p-value to be observed, according to the model described in §3.3.1, we have in general

$$Pr(PV = \varphi) = \sum_{u \in TS^{-1}(PV^{-1}(\varphi))} f_V(u) \tag{3.13}$$

More in detail, referring to Figure 3.6, we note that for each $i, 1 \leq i \leq N_T$, the probability that τ_i is selected (that is, $TS = \tau_i$) as the sample is extracted from U (according to the data distribution f_V) is by construction equal to the sum of the probabilities of the samples mapped on τ_i.

If we define U_i as the counter-image of τ_i (as anticipated in §3.3.1), that is,

$$U_i = TS^{-1}(\tau_i) = \{u \in U | TS(u) = \tau_i\} \tag{3.14}$$

then

$$Pr(TS = \tau_i) = P_i \tag{3.15}$$

with

$$P_i = Pr(u \in U_i) = \sum_{u \in U_i} f_V(u) \tag{3.16}$$

Because of the bijection between T and Φ[14], we have that $Pr(PV = \varphi_i) = Pr(TS = \tau_i)$ and therefore the probability that φ_i is selected (that is, $PV = \varphi_i$) as the sample is extracted from U is again given by (compare with Equation (3.15))

$$Pr(PV = \varphi_i) = P_i \tag{3.17}$$

Replacing $Pr(PV = \varphi_i)$ with $f_{PV}(\varphi_i)$ in Equation (3.17), from Equation (3.6) we can express P_i in terms of the p-values $\{\varphi_i\}$

$$\begin{aligned} P_1 &= \varphi_1 \\ P_i &= \varphi_i - \varphi_{i-1}, i > 1 \end{aligned} \tag{3.18}$$

and vice versa

$$\varphi_i = \sum_{j=1}^{i} P_j, \forall i \tag{3.19}$$

From Equations (3.17), (3.16) and (3.14) we also see that the probability of each p-value φ_i to be observed is determined by the combination of the data distribution f_V and the test statistic function TS because the probability of a sample u to be in the set U_i is determined by f_V, see Equation (3.16), and the set U_i is defined in terms of TS, see Equation (3.14).

We highlight however that in Equation (3.14), for any given i the definition of the U_i set is irrespective of the actual value of τ_i: what is instead relevant is how the test statistic function TS groups the samples of U in subsets $\{U_i\}$ such that for any i all the samples belonging to U_i are mapped on the same test statistic value τ_i. The subsets U_i can, hence, be characterized by the following observations.

Observation 8. *Each U_i is an equivalence class induced by the equivalence relation on the samples defined as* having the same image under the test statistic function.

Observation 9. *U_i does not depend on the output values of the test statistic function.*

In summary, what a test statistic function does is to partition the sample space U in N_Ω subsets (U_i). It also assigns a value τ_i to each subset, which however has the only role to determine an index for the corresponding subset according to its ranking in the tuple of the test statistic values ($i < j$ if and only if $\tau_i < \tau_j$, according to Equation (3.11)). We emphasize that the definition of the test statistic function is

[14] For an analysis of the mentioned bijection, see Footnote 13 at page 42.

part of the definition of the hypothesis test, but the actual values of the test statistic do not impact on the resulting p-values which represent in fact the ultimate output of the test.

This is represented for example in Figure 3.7, where the probability of the orange-circled p-value φ_9 to be observed is equal to the sum of the probabilities (determined by f_V) of the three yellow-circled samples (forming collectively the set $U_9 = TS^{-1}(\tau_9)$) to be extracted from the sample space but does not depend on the actual test statistic value τ_9 (compare with Equations (3.15) and (3.16)):

$$Pr(PV = \varphi_9) = \sum_{u \in U_9} f_v(u)$$

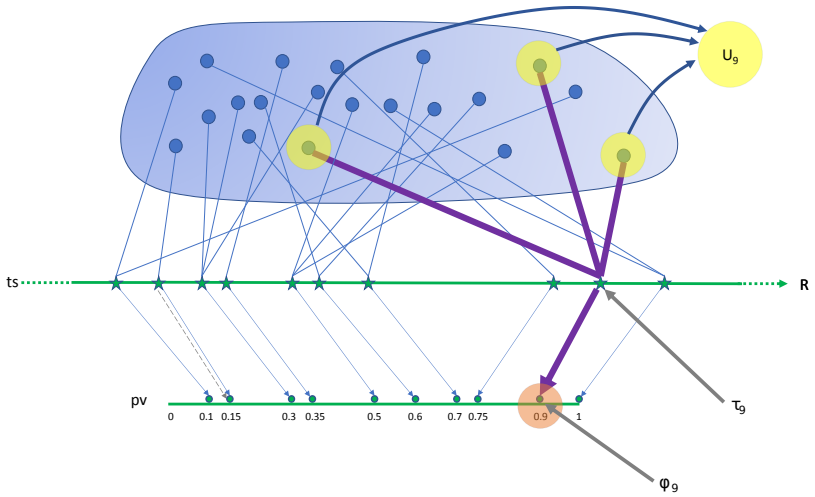


Fig. 3.7 Probability of a p-value

In order to summarize what above discussed, we observe that each test on a sample space U is characterized by three tuples with the same cardinality, given by the number of distinct p-value, N_T: a tuple of subsets $(U_1, U_2, \ldots, U_{N_T})$, a tuple of test statistic values $(\tau_1, \tau_2, \ldots, \tau_{N_T})$ and a tuple of p-values $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_T})$.

3.3.2.1 Ordered set partitions

We note that, given a test on a sample space U, the associated tuple of subsets (U_i) determined by the test as by Observation 8 is an *ordered set partition*, as per the following definition (see also [19]).

Definition 5. *Given a set U, an ordered set partition of U is a list of pairwise disjoint non-empty subsets of U such that the union of these subsets is U.*

We remark that by definition of *list* the order of the subsets is relevant, thus for example (U_1, U_2, U_3) is a different ordered set partition with respect to (U_2, U_3, U_1).

3.3.3 Unconstrained data distribution

Now we analyse the form that a p-tuple can assume when no restriction is set on the data distribution. For this purpose, we first propose the following definition

Definition 6. *Given a sample space U (of cardinality N_U), a tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ is said* valid *if*

1. $N_\Omega \leq N_U$
2. $0 < \omega_i < 1$ *if* $i < N_\Omega$
3. $\omega_{N_\Omega} = 1$
4. $\omega_i < \omega_j$ *if* $i < j$

The first property implies that the number of observable p-values cannot exceed the number of possible samples but can be smaller. The second and third properties mean that the set of the p-values is made of a certain number of values in $(0,1)$ plus an additional maximum value equal to 1. The fourth property requires that the p-values are ordered according to their index.

Recalling Observation 7, which states that a test is defined by the sample space, its data distribution and the test statistic function, we can now express the following theorem

Theorem 2. *Given a sample space, an arbitrary data distribution and an arbitrary test statistic definition, the resulting p-tuple is* valid *(that is, satisfies Definition 6). Vice versa, given a sample space, an arbitrary* valid *tuple represents the p-tuple resulting from at least one choice of the data distribution and of the test statistic function.*

Proof. Given a data distribution and a test statistic function, the set Ω of observable p-values (see Figure 3.6) is defined as

$$\Omega = \{PV(TS(u)), u \in U\}$$

We observe that a such Ω is *valid* according to Definition 6. In particular, the first property is satisfied by construction (with the equality holding when no samples collide on the same test statistic and thus on the same p-value). The second and third properties are satisfied by definition of the p-value as probability and by Observation 3. The fourth property can be assumed true without loss of generality, as it is implicit in the definition of p-tuple (see §3.2.1).

Conversely, given a sample space $U = \{u_1, u_2, \ldots, u_{N_U}\}$ and an arbitrary tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ satisfying Definition 6, we can choose a data distribution f_V and a test statistic function TS producing a p-tuple equal to Ω according to Equation (3.13) as follows.

First we define a tuple $T = (\tau_1, \tau_2, \ldots, \tau_{N_\Omega})$, made of N_Ω distinct ordered real numbers (their actual values are irrelevant, it is just required that $\tau_i < \tau_j$ for $i < j$). Then we

- define the data distribution f_v as

$$f_V : U \to [0,1] \tag{3.20}$$
$$f_V(u_i) = \begin{cases} \omega_1 & \text{if } i = 1 \\ \omega_i - \omega_{i-1} & \text{if } i \in [2, N_\Omega - 1] \\ \dfrac{1 - \omega_{N_\Omega - 1}}{N_U - N_\Omega + 1} & \text{if } i \in [N_\Omega, N_U] \end{cases}$$

  which is consistently defined since

  - $f_V(u) > 0, \forall u \in U$
  - $\sum_{u \in U} f_v(u) = 1$

  because of properties 2 and 4 of Definition 6;

- define the test statistic function TS

$$TS : U \to T \tag{3.21}$$
$$TS(u_i) = \begin{cases} \tau_i & \text{if } i \in [1, N_\Omega - 1] \\ \tau_{N_\Omega} & \text{if } i \in [N_\Omega, N_U] \end{cases}$$

- observe that the p-value function PV is implicitly defined as[15]

$$PV : T \to \Omega$$
$$PV(\tau_i) = \omega_i \qquad \forall i \in [1, N_\Omega]$$

What we do, in essence, is to build a one-to-one map from the sample space U to the set T for all but one elements τ_i of T and to map all the remaining samples of U onto the last element of T, τ_{N_Ω}. In doing so we normalize the samples probabilities used in the second step in order to obtain the desired probability for τ_{N_Ω} independently of the number of samples mapped on τ_{N_Ω} (which is guaranteed to be always at least one from property 1 of Definition 6).

Referring to the definition of the sets U_i and of the probabilities P_i given by Equations (3.14) and (3.16), and here recalled in the following equations:

$$U_i = \{u \in U | TS(u) = \tau_i\}$$

$$P_i = Pr(u \in U_i)$$

we observe that the above-described process determines N_Ω sets U_i ($i = 1, 2, \ldots, N_\Omega$) as follows.

For any $i \in [1, N_\Omega - 1]$ the set U_i is defined by a unique sample $U_i = \{u_i\}$ with

$$\begin{aligned} P_1 &= \omega_1 \\ P_i &= \omega_i - \omega_{i-1}, \text{ if } i \in [2, N_\Omega - 1] \end{aligned} \tag{3.22}$$

[15] According to the model depicted in Figure 3.6 we also need to define a p-value function PV but, as summarized in Observation 7, PV is entirely determined by f_V and TS.

The last set U_{N_Ω} is composed by the remaining $N_U - N_\Omega + 1$ samples, $U_{N_\Omega} = \{u_j, j \in [N_\Omega, N_U]\}$, with

$$P_{N_\Omega} = \omega_{N_\Omega} - \omega_{N_\Omega - 1} \tag{3.23}$$

Comparing Equations (3.22) and (3.23) with Equation (3.18), we derive that τ_i is in turn mapped onto the p-value ω_i for each i, thus concluding the proof.

□

The whole construction is exemplified in Figure 3.8, with $N_U = 8$ and $N_\Omega = 6$, where $\Omega = (\omega_1, \omega_2, \ldots, \omega_6)$ is the target p-tuple we want to obtain[16].

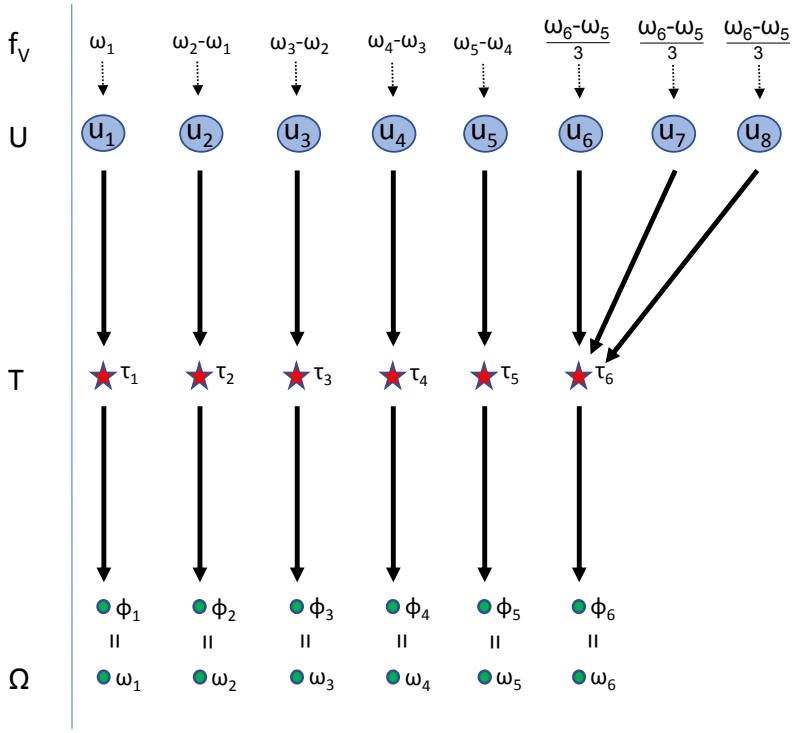


Fig. 3.8 Construction of an arbitrary Ω set

Recalling from Observation 7 that a test is completely defined by a triple made of the sample space, its data distribution and the test statistic function, the meaning of Theorem 2 is that all and only the *valid* p-tuples represent legitimate tests (that

[16]We observe that the pair of maps f_V and TS defined in Equations (3.20) and (3.21) is just one among infinitely many, with the exception of the case $N_U = N_\Omega$ where the solution is essentially unique (up to permutations of the samples in the definition of the subsets $\{U_i\}$).

is, consistently defined by a sample space, a data distribution and a test statistic function, as for Definition 4).

3.3.4 Fixed data distribution

Now we consider the case in which the data distribution is (arbitrary but) fixed, that is, the underlying distribution of the sample extraction process is unique, thus defining a *simple* null hypothesis (see §2.1)[17].

3.3.4.1 Equivalent tests

We first propose the following definition, which turns out to be fundamental in the fixed data distribution setting.

Definition 7. *Given a sample space and its data distribution, two tests are said* equivalent *if, given an arbitrary identical input, they produce an identical p-value as output*[18].

Two equivalent tests are, hence, essentially the same test (and, consequently, produce the same p-tuple). Since it makes sense to compare two tests only under the same data distribution, in the following, when we refer to equivalent tests, we always (sometimes implicitly) assume they have the same data distribution.

We are now able to state the following theorem.

Theorem 3. *Given two tests on a sample space with the same data distribution, they are equivalent if and only if they determine the same ordered set partition.*

Proof. If the two tests determine the same ordered set partition (U_i) (see §3.3.2.1), then, given a sample u it belongs for both tests to the same subset of the sample space U, say U_i for some index i. Then by construction (see Figure 3.6) the sample u

[17]The setting considered includes the one required to build a hypothesis test for randomness, where the sample space is made of all the possible binary sequences of a given length and the null hypothesis is that they are independently and uniformly extracted.

[18]We remind from §2.2.3.2 that the most convenient way to interpret a hypothesis test is that it produces in output a probability value (the p-value). A further value α, which is compared with the resulting probability to take a hard acceptance/rejection decision about the null hypothesis, can be seen as an external parameter.

is sent on two test statistics $\tau 1_i$ and $\tau 2_i$, possibly distinct for the two tests but with the same index i. Finally from $\tau 1_i$ and $\tau 2_i$ it is sent on two values $\varphi 1_i, \varphi 2_i$, with $\varphi 1_i = \varphi 2_i$ because of equations (3.12), (3.15) and (3.16). Since this holds for each sample u, then the two tests are equivalent by Definition 7.

Conversely, if the two tests are equivalent, then each u is sent for both tests to the the same p-value, say φ_i for some index i. This means that the corresponding test statistic values for the two tests, despite being possibly different, share the same index i: $\tau 1_i$ and $\tau 2_i$. Then, by Equation (3.14), u belongs to the same subset U_i for both tests, again for the same index i. Since this holds for each sample u, it follows that the two ordered set partitions, corresponding to the two tests, coincide. □

Theorem 3 states that tests can be grouped in disjoint classes, where each class is made of all and only the equivalent tests, while (consequently) non-equivalent tests belong to different classes. Moreover, each class is associated to an ordered set partition, which can thus be interpreted as the essential representation of the whole class. Then, with a little abuse of terminology, we can restate Theorem 3 as follows

Observation 10. *There exists a bijection between ordered set partitions and non-equivalent tests.*

It is worth pointing out that the equivalence among tests is irrespective of the actual values of the test statistic tuples, as can be shown extending Observation 9 from a single subset U_i to the whole ordered set partition. Hence, for a fixed ordered set partition, any set of test statistic values defines a different but equivalent test. Thus, since there are infinite choices for the test statistic values, we can make the following

Observation 11. *Each test defines (and belongs to) a class of infinite equivalent tests, which differ for the output values of the test statistic function.*

3.3.4.2 F-valid tuples

We can now define the concept of *F-valid* tuples

Definition 8. *Given a sample space* $U = \{u_1, u_2, \ldots, u_{N_U}\}$ *and its data distribution* $\{f_V(u_i), i \in [1, N_U]\}$*, a tuple* $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ *is said* F-valid *if there exists an*

ordered set partition M of U

$$M = (U_1, U_2, \ldots, U_{N_\Omega})$$

such that

$$\omega_i = \sum_{j=1}^{i} P_j, \forall i \in [1, N_\Omega]$$

with

$$P_j = \sum_{u \in U_j} fv(u)$$

We also say that Ω *is the p-tuple associated to M and we denote it by* Ω_M.

A graphical representation of the meaning of Definition 8 is given in Figure 3.9, where $N_U = 8, N_\Omega = 4$ and $M = (U_1, U_2, U_3, U_4)$ with

$$U_1 = \{u_1, u_2\}, U_2 = \{u_3\}, U_3 = \{u_4, u_5, u_6, u_7\}, U_4 = \{u_8\}$$

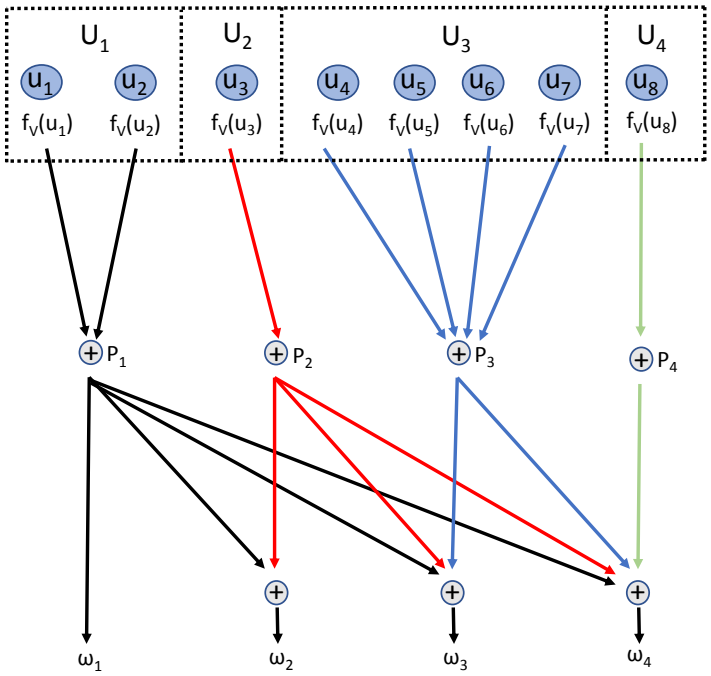


Fig. 3.9 Example of F-valid tuple

By Definition 8 we can make the following observation.

Observation 12. *For an arbitrarily fixed data distribution, the (F-valid) p-tuple associated to a a given ordered set partition is uniquely defined. On the contrary, a single (F-valid) p-tuple can be associated with multiple ordered set partitions or may not be associated with any ordered set partition at all.*

With regard to Observation 12, here we anticipate that Theorem 5 will provide a necessary and sufficient condition to have a single ordered set partition for a given p-tuple (or, equivalently, to have a bijection between the set of ordered set partitions and the set of (F-valid) p-tuples).

Now we can state the following theorem.

Theorem 4. *Given a sample space, a fixed data distribution and an arbitrary test statistic definition, the resulting p-tuple is* F-valid *(that is, satisfies Definition 8). Vice versa, given a sample space and a fixed data distribution, an arbitrary* F-valid *tuple represents the p-tuple resulting from at least one choice of the test statistic function.*

Proof. Given a sample space U of cardinality N_U, a fixed data distribution f_V and a test statistic function TS mapping U on a tuple $T = (\tau_1, \tau_2, \ldots, \tau_{N_T})$, by construction (see Equations (3.14), (3.15), (3.16) and (3.19)) the resulting p-tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ is determined as

$$\omega_i = \sum_{j=1}^{i} P_j,$$

with $N_\Omega = N_T$ and $P_j = \underset{u \in U}{Pr}(u \in TS^{-1}(\tau_j))$.

If we set

$$U_i = TS^{-1}(\tau_i), i \in [1, N_\Omega]$$

then Ω is associated to the ordered set partition $M = (U_1, U_2, \ldots, U_{N_\Omega})$ according to Definition 8 and Ω is therefore F-valid.

Conversely, given a sample space U of cardinality N_U, a fixed data distribution f_V and a F-valid tuple

$$\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$$

we want to choose a test statistic function such that the resulting p-tuple is Ω.

To do so, we observe that Equation (3.13) still holds, but in order to assign the target probability to the p-values probabilities $Pr(PV = \varphi)$ (on the left side of the

equation) we can no more rely on modifications of the data distribution ($f_V(u)$) (on the right side of the equation) which is now fixed. So we are left to play with the summation range only and then with the definition of the test statistic TS as follows.

Since Ω is F-valid we know by Definition 8 that there exists an ordered set partition $M = (U_1, U_2, \ldots, U_{N_\Omega})$ such that

$$\omega_i = \sum_{j=1}^{i} P_j, \forall i \in [1, N_\Omega]$$

with

$$P_j = \sum_{u \in U_j} fv(u)$$

We define an arbitrary tuple $T = (\tau_1, \tau_2, \ldots, \tau_{N_\Omega})$, made of N_Ω arbitrary distinct real numbers with $\tau_i < \tau_j$ for $i < j$. We then set

$$TS(u) = \tau_i, \forall u \in U_i, \forall i \in [1, N_\Omega]$$

By construction the p-tuple determined by the test statistic function above defined coincides with Ω. □

3.3.4.3 On the number of non-equivalent tests

Given a sample space U of cardinality N_U in the fixed data distribution setting, it is interesting to count the number of non-equivalent tests. By Observation 10 we know that non-equivalent tests and ordered set partitions are in bijective relation, so we can instead count the number $B(N_U)$ of distinct ordered set partitions, which can be expressed as in Equation (3.24):

$$B(N_U) = \sum_{k=1}^{N_U} \left\{ {N_U \atop k} \right\} k! \tag{3.24}$$

where $\left\{ {N_U \atop k} \right\}$ represents the Stirling number of the second kind, that is, the number of ways we can partition a set of N_U elements into k non-empty subsets[19]. Since we are taking the order of the parts in account, we still have to multiply that number by

[19] A such number can be expressed as $\left\{ {N_U \atop k} \right\} = \frac{1}{k!} \sum_{i=0}^{k} (-1)^i \binom{k}{i} (k-i)^{N_U}$. See [20].

$k!$ in order to have all the partitions with k parts. Finally we have to span over the possible values of k (from 1 to N_U) to consider all the possible ordered set partitions, thus obtaining $B(N_U)$ (which is known as *Fubini Number* or *Ordered Bell Number*, see [21]).

The first elements of the sequence $(B_{N_U}, N_U \geq 1)$[20] are:

$$1, 3, 13, 75, 541, 4683, 47293, 545835, 7087261, \ldots$$

For example, if $N = 4$ we have the 75 ordered set partitions reported in Table 3.1, where M_k represents the generic partition of U in k subsets.

k	M_k	$\{U_i\}$
1	(U_1)	$\{u_1,u_2,u_3,u_4\}$
2	(U_1,U_2)	$(\{u_1\},\{u_2,u_3,u_4\}), (\{u_2,u_3,u_4\},\{u_1\})$ $(\{u_2\},\{u_1,u_3,u_4\}), (\{u_1,u_3,u_4\},\{u_2\})$ $(\{u_3\},\{u_1,u_2,u_4\}), (\{u_1,u_2,u_4\},\{u_3\})$ $(\{u_4\},\{u_1,u_2,u_3\}), (\{u_1,u_2,u_3\},\{u_4\})$ $(\{u_1,u_2\},\{u_3,u_4\}), (\{u_3,u_4\},\{u_1,u_2\})$ $(\{u_1,u_3\},\{u_2,u_4\}), (\{u_2,u_4\},\{u_1,u_3\})$ $(\{u_1,u_4\},\{u_2,u_3\}), (\{u_2,u_3\},\{u_1,u_4\})$
3	(U_1,U_2,U_3)	$(\{u_1\},\{u_2\},\{u_3,u_4\}), (\{u_1\},\{u_3,u_4\},\{u_2\}), (\{u_2\},\{u_1\},\{u_3,u_4\})$ $(\{u_2\},\{u_3,u_4\},\{u_1\}), (\{u_3,u_4\},\{u_1\},\{u_2\}), (\{u_3,u_4\},\{u_2\},\{u_1\})$ $(\{u_1\},\{u_3\},\{u_2,u_4\}), (\{u_1\},\{u_2,u_4\},\{u_3\}), (\{u_3\},\{u_1\},\{u_2,u_4\})$ $(\{u_3\},\{u_2,u_4\},\{u_1\}), (\{u_2,u_4\},\{u_1\},\{u_3\}), (\{u_2,u_4\},\{u_3\},\{u_1\})$ $(\{u_1\},\{u_4\},\{u_2,u_3\}), (\{u_1\},\{u_2,u_3\},\{u_4\}), (\{u_4\},\{u_1\},\{u_2,u_3\})$ $(\{u_4\},\{u_2,u_3\},\{u_1\}), (\{u_2,u_3\},\{u_1\},\{u_4\}), (\{u_2,u_3\},\{u_4\},\{u_1\})$ $(\{u_2\},\{u_3\},\{u_1,u_4\}), (\{u_2\},\{u_1,u_4\},\{u_3\}), (\{u_3\},\{u_2\},\{u_1,u_4\})$ $(\{u_3\},\{u_1,u_4\},\{u_2\}), (\{u_1,u_4\},\{u_2\},\{u_3\}), (\{u_1,u_4\},\{u_3\},\{u_2\})$ $(\{u_2\},\{u_4\},\{u_1,u_3\}), (\{u_2\},\{u_1,u_3\},\{u_4\}), (\{u_4\},\{u_2\},\{u_1,u_3\})$ $(\{u_4\},\{u_1,u_3\},\{u_2\}), (\{u_1,u_3\},\{u_2\},\{u_4\}), (\{u_1,u_3\},\{u_4\},\{u_2\})$ $(\{u_3\},\{u_4\},\{u_1,u_2\}), (\{u_3\},\{u_1,u_2\},\{u_4\}), (\{u_4\},\{u_3\},\{u_1,u_2\})$ $(\{u_4\},\{u_1,u_2\},\{u_3\}), (\{u_1,u_2\},\{u_3\},\{u_4\}), (\{u_1,u_2\},\{u_4\},\{u_3\})$
4	(U_1,U_2,U_3,U_4)	$(\{u_1\},\{u_2\},\{u_3\},\{u_4\}), (\{u_1\},\{u_2\},\{u_4\},\{u_3\}), (\{u_1\},\{u_3\},\{u_2\},\{u_4\})$ $(\{u_1\},\{u_3\},\{u_4\},\{u_2\}), (\{u_1\},\{u_4\},\{u_2\},\{u_3\}), (\{u_1\},\{u_4\},\{u_3\},\{u_2\})$ $(\{u_2\},\{u_1\},\{u_3\},\{u_4\}), (\{u_2\},\{u_1\},\{u_4\},\{u_3\}), (\{u_2\},\{u_3\},\{u_1\},\{u_4\})$ $(\{u_2\},\{u_3\},\{u_4\},\{u_1\}), (\{u_2\},\{u_4\},\{u_1\},\{u_3\}), (\{u_2\},\{u_4\},\{u_3\},\{u_1\})$ $(\{u_3\},\{u_1\},\{u_2\},\{u_4\}), (\{u_3\},\{u_1\},\{u_4\},\{u_2\}), (\{u_3\},\{u_2\},\{u_1\},\{u_4\})$ $(\{u_3\},\{u_2\},\{u_4\},\{u_1\}), (\{u_3\},\{u_4\},\{u_1\},\{u_2\}), (\{u_3\},\{u_4\},\{u_2\},\{u_1\})$ $(\{u_4\},\{u_1\},\{u_2\},\{u_3\}), (\{u_4\},\{u_1\},\{u_3\},\{u_2\}), (\{u_4\},\{u_2\},\{u_1\},\{u_3\})$ $(\{u_4\},\{u_2\},\{u_3\},\{u_1\}), (\{u_4\},\{u_3\},\{u_1\},\{u_2\}), (\{u_4\},\{u_3\},\{u_2\},\{u_1\})$

Table 3.1 List of all the ordered set partitions for set size equal to 4

[20]The sequence $(B_{N_U}, N_U \geq 1)$ is indexed as A000670 in the On-Line Encyclopedia of Integer Sequences (OEIS) [22]. For the sake of precision, the mentioned sequence in OEIS includes also the first element B_0=1.

3.3.4.4 On the number of observable p-tuples

We also note that Theorem 4 states that all and only the F-valid tuples are observable p-tuples and by Observation 12 we derive that $B(N_U)$ is also an upper limit for the number of distinct F-valid tuples and thus of distinct observable p-tuples. The actual number of distinct observable p-tuples depends on the data distribution f_V of the sample space U, as shown in the following theorem, which determines when this upper limit is reached (that is, there are no ordered set partitions colliding on the same F-valid tuple).

Theorem 5. *Given a set U and its $(B(N_U))$ distinct ordered set partitions, the associated p-tuples are all distinct if and only if there is no pair of subsets $U1, U2 \subset U$, with $U1 \neq U2$ and $P1 = P2$, where $P1$ and $P2$ are the subset probabilities defined as $P1 = \sum_{u \in U1} f_V(u), P2 = \sum_{u \in U2} f_V(u)$.*

Proof. Let us assume there exist two subsets $U1, U2$, with $U1 \neq U2$ and $P1 = P2$. Then the two ordered set partitions

$$M = (U1, U2, U \setminus (U1 \cup U2))$$

and

$$M' = (U2, U1, U \setminus (U1 \cup U2))$$

produce the same p-tuple

$$\Omega_M = \Omega_{M'} = (P1, 2P1, 1)$$

Conversely, let us take two ordered set partitions

$$M = (U_1, U_2, \ldots, U_k)$$

and

$$M' = (U'_1, U'_2, \ldots, U'_k)$$

with the corresponding p-tuples

$$\Omega_M = (\omega_1, \omega_2, \ldots, \omega_k)$$

and

$$\Omega_{M'} = (\omega'_1, \omega'_2, \ldots, \omega'_k)$$

and let us suppose that $M \neq M'$ but $\Omega_M = \Omega_{M'}$.

Since $M \neq M'$, there are one (in fact, at least two) or more indexes such that the corresponding subset in the two partitions differ. Let i^* be the smallest such index:

$$\begin{aligned} U_i &= U'_i, \forall i < i^* \\ U_{i^*} &\neq U'_{i^*} \end{aligned} \tag{3.25}$$

Since $\Omega_M = \Omega_{M'}$, we have $\forall i, \omega_i = \omega'_i$ and in particular $\omega_{i^*} = \omega'_{i^*}$ or equivalently, by Definition 8,

$$\sum_{i=1}^{i^*} P_i = \sum_{i=1}^{i^*} P'_i \tag{3.26}$$

with $P_i = \sum_{u \in U_i} f_V(u), P'_i = \sum_{u \in U'_i} f_V(u)$.

Since $U_i = U'_i, \forall i < i^*$ we also have $P_i = P'_i, \forall i < i^*$. Therefore, in order for Equation (3.26) to hold, it is necessary that

$$P_{i^*} = P'_{i^*} \tag{3.27}$$

Comparing Equations (3.25) and (3.27) we see that we have found two distinct subsets with the same subset probability. □

Hence, Theorem 5 gives a necessary and sufficient condition to obtain the maximum number $B(N_U)$ of distinct p-tuples (given by Equation (3.24)) as the ordered set partition takes all the possible configurations (which, by Observation 10, is equivalent to say that all possible non-equivalent tests are considered). The condition is expressed in terms of the data distribution of the sample space U of cardinality N_U.

3.3.4.5 An example

As an example, consider a set $U = \{u_1, u_2, \ldots, u_6\}$ of cardinality 6, with the following data distribution

$$f_V = \{0.05, 0.25, 0.20, 0.18, 0.22, 0.10\}$$

Then we have

$$B_6 = \sum_{k=1}^{6} \left\{ {6 \atop k} \right\} k! = \\ 1 \cdot 1 + 31 \cdot 2 + 90 \cdot 6 + 65 \cdot 24 + 15 \cdot 120 + 1 \cdot 720 = 4683 \tag{3.28}$$

Now, since $f_V(u_1) + f_V(u_2) + f_V(u_6) = f_V(u_4) + f_V(u_5)$, as shown in Theorem 4 we can build two ordered set partitions of U with the same resulting p-tuples:

$$M1 = (\{u_1, u_2, u_6\}, \{u_4, u_5\}, \{u_3\})$$

$$M2 = (\{u_4, u_5\}, \{u_1, u_2, u_6\}, \{u_3\})$$

with

$$\Omega_{M1} = \Omega_{M2} = (0.4, 0.8, 1)$$

Hence, the total number of different p-tuples is less than 4683, which is the total number of possible ordered set partitions computed in Equation (3.28).

We note however that it is always possible to define a data distribution such that all the p-tuples are distinct, that is, each (non-equivalent) test on the sample space U determines a different p-tuple. For example the following data distributions

$$\forall b > 1, k \in \mathbb{N}, \left\{ f_V(u_i) = \frac{b-1}{b^k - 1} b^{i-1}, i = 1, 2, \ldots, k \right\} \tag{3.29}$$

with $k = N_U$ for an arbitrary set U of cardinality N_U, guarantee that there are no subsets with the same probability (since b can be seen as the base of the b-ary numeral system, all the sums of powers of b are distinct).

For instance, in the same setting of the previous example ($N_U = 6$) we can build the following data distribution according to Equation (3.29) with $b = 2$, guaranteeing by construction that all the 4683 resulting p-tuples are different

$$f_V = \left\{ \frac{1}{63}, \frac{2}{63}, \frac{4}{63}, \frac{8}{63}, \frac{16}{63}, \frac{32}{63} \right\}$$

3.3.4.6 A graphical representation

Figure 3.10 gives an illustrative view of the relation among (equivalent) tests, ordered set partitions and F-valid p-tuples. In particular, tests are shown grouped by equivalence sets (in the upper area of the figure), which are mapped onto the corresponding ordered set partitions (central area), which in turn are mapped onto F-valid tuples (shown in the lower area). We note that tests are infinite (see Observation 11), while ordered set partitions (that is, non-equivalent tests) and F-valid tuples are finite, as shown in §3.3.4.3 and §3.3.4.4.

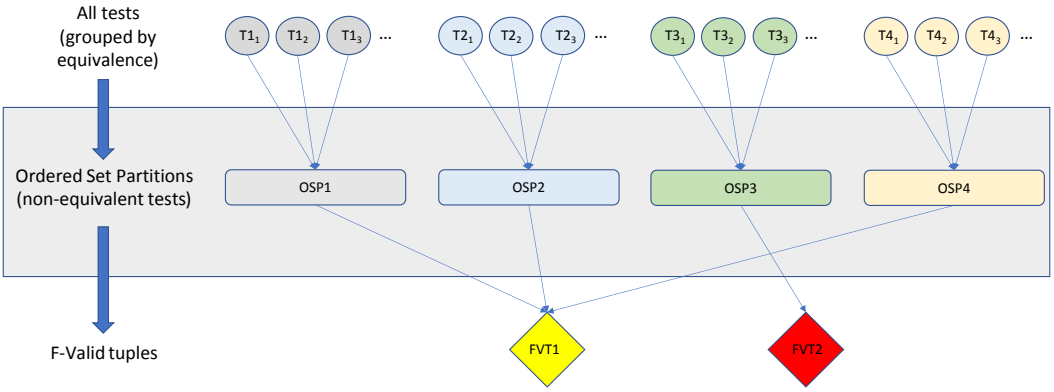


Fig. 3.10 Relation among tests, ordered set partitions and F-valid tuples (I)

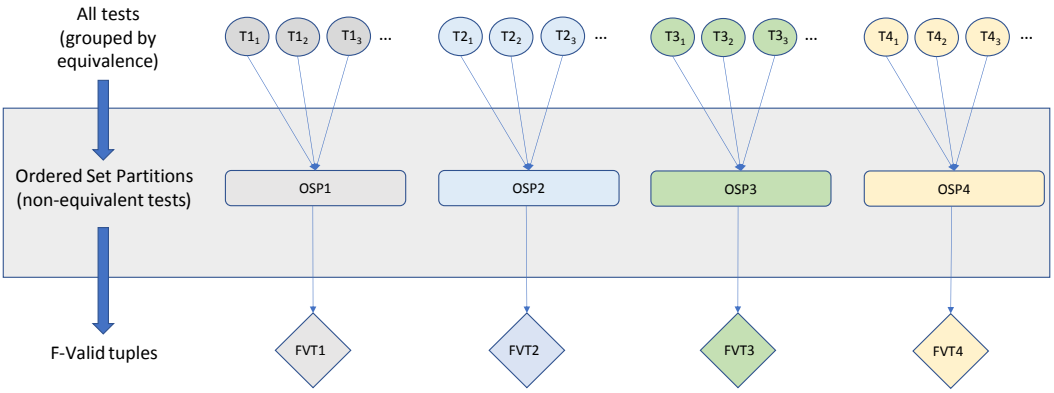


Fig. 3.11 Relation among tests, ordered set partitions and F-valid tuples (II)

Figure 3.11 adapts the generic scheme of Figure 3.10 to the case where, according to Theorem 5, all the p-tuples are distinct and are, thus, in bijective relations with the ordered set partitions.

3.3.5 Uniform data distribution

A specific setting is when the data distribution is the uniform one. Being in the finite case, we observe that a finite distribution cannot be uniform in the continuous sense. A useful approximation is however represented by the *discrete uniformity* property, which requires that all the possible outcomes have the same probability

Definition 9. *A random variable is said* discrete uniform *(or to have a* discrete uniform *distribution) if all the possible outcomes are equally likely to be observed.*

In our case it means that, given the sample space U of cardinality N_U, all the samples $u \in U$ have the same probability $\frac{1}{N_U}$ to be observed. This is of particular interest for us because it is exactly what we consider when applying statistical tests for randomness, where we assume that the random generator under analysis produces equally likely samples (that is, binary sequences of a given length).

In this context it is useful to introduce the following definition

Definition 10. *Given a sample space $U = \{u_1, u_2, \ldots, u_{N_U}\}$ with discrete uniform data distribution, a tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ is said* U-valid *if it can be expressed in the form*

$$\Omega = (\frac{k_1}{N_U}, \frac{k_2}{N_U}, \ldots, \frac{k_{N_\Omega - 1}}{N_U}, 1) \tag{3.30}$$

with

- $N_\Omega \leq N_U$*;*
- k_i *integer* $\in [1, N_U - 1]$, $\forall i \in [1, N_\Omega - 1]$*;*
- $k_i < k_{i+1}, \forall i \in [1, N_\Omega - 2]$*;*
- $k_{N_\Omega - 1} < N_U$.

We can prove the following theorem, which states that Equation (3.30) gives the form of all and only the *F-valid* tuples in the uniform distribution case.

Theorem 6. *Under the data distribution discrete uniformity assumption, a tuple is U-valid if and only if it is F-valid.*

Proof. Let us assume that $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega - 1}, \omega_{N_\Omega})$ is a F-valid tuple. Then by Definition 8 there exists an ordered set partition

$$M = (U_1, U_2, \ldots, U_{N_\Omega})$$

with $N_\Omega \leq N_U$ and such that, for any $i \in [1, N_\Omega]$,

$$\omega_i = \sum_{j=1}^{i} P_j, \forall i \in [1, N_\Omega] \tag{3.31}$$

with

$$P_j = \sum_{u \in U_j} f_V(u)$$

By definition of discrete uniformity, we have

$$f_V(u) = \frac{1}{N_U}, \forall u \in U$$

and then

$$P_j = \frac{|U_j|}{N_U}$$

The tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega - 1}, \omega_{N_\Omega})$ defined by Equation (3.31) then satisfies Definition 10 and is thus U-valid.

Conversely, let us assume that $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega - 1}, \omega_{N_\Omega})$ is a U-valid tuple. Then Ω is associated to any ordered set partition $M = (U_1, U_2, \ldots, U_{N_\Omega})$ such that

$$|U_1| = k_1, |U_2| = k_2 - k_1, \ldots, |U_{N_\Omega - 1}| = k_{N_\Omega - 1} - k_{N_\Omega - 2}, |U_{N_\Omega}| = N_U - k_{N_\Omega - 1}$$

Since

$$\sum_{i=1}^{N_\Omega} |U_i| = N_U$$

it is always possible to define a such M and then by Definition (8) Ω is a F-valid tuple. □

Thanks to Theorem 6 we can now adapt Theorem 4 to the discrete uniform data distribution case:

Theorem 7. *Given a sample space with discrete uniform data distribution and an arbitrary test statistic definition, the resulting p-tuple is always* U-valid *(that is,*

satisfies Definition 10). Vice versa, given a sample space with discrete uniform data distribution, an arbitrary U-valid *tuple represents the p-tuple resulting from at least one choice of the test statistic function.*

Let us now briefly compare Theorem 2 and Theorem 7. Theorem 2 essentially states that, when the data distribution is not fixed, any *valid* tuple of distinct real values in $(0,1]$ (that is, any strictly increasing sequence which includes 1 as last value, see Definition 6) corresponds to a legitimate test (see Definition 4) and vice versa. Theorem 7 states that the same holds when the data distribution is fixed and uniform, with the more stringent constraint that the tuple of real values is U-valid (see Definition 10).

We remark from Theorem 7 that *any* Ω which is U-valid corresponds to a legitimate test (see Definition 4), that is, there exists a test statistic function (in fact infinitely many, simply playing with the test statistic values, according to Observation 11) that, given the discrete uniform data distribution, determines Ω as p-tuple. This is a fundamental fact because it means that if we do not have knowledge about the test statistic function, then we cannot make any assumption on the associated p-tuple, apart from being in the form given by Equation (3.30).

3.3.5.1 An example

As an example of application of the theory developed in the previous sections, we now consider the setting where the sample space U is made of all the 4-bit sequences (and then $N_U = 16$) and the data distribution is discrete uniform (that is, our null hypothesis is that all the sequences are equally likely). If we consider the p-tuple

$$\Omega = (\omega_1, \omega_2, \omega_3, \omega_4, \omega_5) = (\frac{3}{16}, \frac{5}{16}, \frac{11}{16}, \frac{14}{16}, 1)$$

then we observe that Ω is a U-valid tuple, according to Definition 10. Hence, by Theorem 6 we know that it is F-valid as well and thus, by Definition 8, that there is at least one corresponding ordered set partition. In fact, we can choose any ordered set partition of U, $M = (U_1, U_2, U_3, U_4, U_5)$ such that

$$|U_1| = 3, |U_2| = 2, |U_3| = 6, |U_4| = 3, |U_5| = 2$$

Examples of solutions are

$$U_1 = \{u_1, u_2, u_3\}, U_2 = \{u_4, u_5\}, U_3 = \{u_6, u_7, u_8, u_9, u_{10}, u_{11}\},$$
$$U_4 = \{u_{12}, u_{13}, u_{14}\}, U_5 = \{u_{15}, u_{16}\}$$

and also any other choice of the samples which maintains the cardinality of the subsets ($|U_i|$), like for example

$$U_1 = \{u_4, u_7, u_{15}\}, U_2 = \{u_2, u_9\}, U_3 = \{u_1, u_3, u_{10}, u_{13}, u_{14}, u_{16}\},$$
$$U_4 = \{u_5, u_8, u_{12}\}, U_5 = \{u_6, u_{11}\}$$

Once an ordered set partition is built as above described, a legitimate test can be obtained determining Ω as p-tuple (see proof of Theorem 4).

From Theorem 7 we also know that p-tuples which are not U-valid are instead impossible to obtain. For example, the p-tuple

$$\Omega = (\omega_1, \omega_2, \omega_3, \omega_4, \omega_4, \omega_6) = (\frac{4}{32}, \frac{7}{32}, \frac{14}{32}, \frac{19}{32}, \frac{26}{32}, 1)$$

is impossible to observe, because ω_2 and ω_4 cannot be put in the form $\frac{k}{16}$ with k integer (see Definition 10).

Conversely, in Theorem 7 we have also shown that any U-valid tuple actually corresponds to a legitimate test. Below we verify this claim with the following trivial U-valid p-tuple:

$$\Omega = (\omega_1, \omega_2) = (\frac{15}{16}, 1)$$

which is really poor in terms of uniformity, as clearly shown also by the PMF in Figure 3.12.

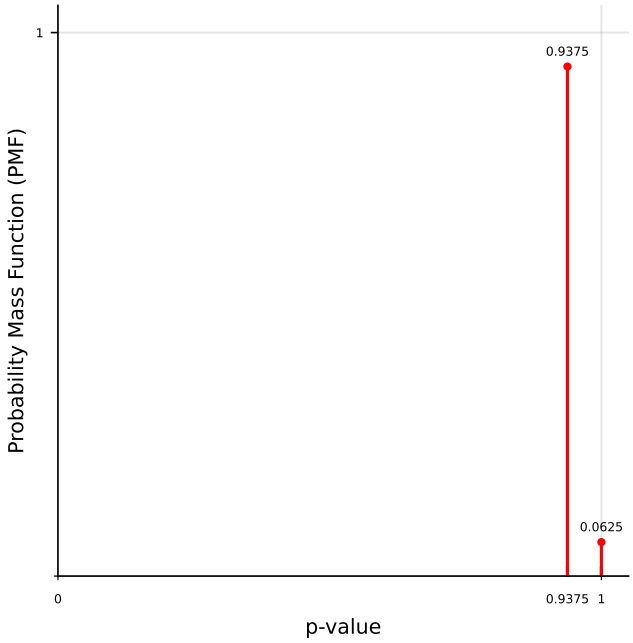


Fig. 3.12 Poorly uniform p-tuple

The given Ω is, however, the p-tuple of the legitimate test defined through its test statistic function:

$$TS(u_i) = \tau_1, \forall i < 16$$
$$TS(u_{16}) = \tau_2$$

for arbitrary τ_1, τ_2 with $\tau_1 < \tau_2$.

3.3.6 Section synthesis

Here we briefly recapitulate the concepts introduced and developed in the current Section §3.3. In particular, in §3.3.6.1 we recall the general notions used throughout the whole section, while in the following Subsections §3.3.6.2, §3.3.6.3 and §3.3.6.4 we focus on the three data distribution settings considered, namely the unconstrained one, the fixed one and finally the uniform one. Results reported apply to the specific setting considered in the corresponding subsection.

3.3.6.1 General

See §3.3.1 and §3.3.2.

- It is shown that each test is determined by a triple (U, f_V, TS), where U is the sample space, f_V its data distribution and finally TS the test statistic function;
- the concept of *legitimate test* is introduced, to identify tests whose triple is consistently given;
- an ordered set partition $(U_1, U_2, \ldots, U_{N_T})$ is associated to each test, where N_T is the number of distinct values produced by the test statistic function as the sample varies in the sample space U;
- it is shown that each U_i is an equivalence class made of the set of samples mapped on the same test statistic value;
- a p-tuple is associated to each ordered set partition and, thus, to each test. It is made of the ordered set of p-values produced by the test as the sample varies in the sample space U according to the data distribution f_V;
- it is proved that the p-tuple associated to a a given ordered set partition is uniquely defined. On the contrary a single p-tuple can be associated to more (or no) ordered set partitions.

3.3.6.2 Unconstrained data distribution setting

See §3.3.3.

- The concept of *valid* p-tuple is introduced and characterized;
- it is shown that all and only the valid p-tuples correspond to legitimate tests.

3.3.6.3 Fixed data distribution setting

See §3.3.4.

- The concept of *equivalent tests* is introduced, that is, two tests that, under the same data distribution, associate the same p-value to the same sample;

- it is proven that two tests are equivalent if and only if they determine the same ordered set partition;
- it is shown that infinite equivalent tests are associated to each ordered set partition and that equivalent tests differ only in the values produced by the test statistic function (but samples are grouped in the same ordered set partition);
- the concept of F-valid p-tuple is introduced and characterized;
- it is shown that all and only the F-valid p-tuples correspond to legitimate tests;
- it is proved that there exists a bijection between the set of the ordered set partitions and the set of the non-equivalent tests;
- the number of non-equivalent tests is determined;
- a necessary and sufficient condition to have different p-tuples for different non-equivalent tests is derived, depending on the underlying data distribution.

3.3.6.4 Uniform data distribution setting

See §3.3.5.

- The concept of U-valid p-tuple is introduced and characterized;
- it is shown that all and only the U-valid p-tuples correspond to legitimate tests.

3.4 On the uniformity of p-values

Specializing to the p-value random variable the general concept of discrete uniformity given with Definition 9, we have the following definition:

Definition 11. *The random variable PV with support* $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ *is said* discrete uniform *if* $Pr(PV = \omega_i) = \dfrac{1}{N_\Omega}, \forall i \in [1, N_\Omega]$

The following theorem provides a necessary and sufficient condition for the discrete uniformity of the p-values.

Theorem 8. *The random variable PV with support* $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ *is discrete uniform if and only if its elements are uniformly spaced and more precisely:*

$$Pr(PV = \omega_i) = \frac{1}{N_\Omega}, \forall i \in [1, N_\Omega]$$

if and only

$$\omega_i = \frac{i}{N_\Omega}, \forall i \in [1, N_\Omega] \tag{3.32}$$

Proof. The theorem is a direct consequence of Equation (3.6). □

We point out that Equation (3.3), despite stating a (weak) form of uniformity even in the discrete distribution case, in general does not provide *discrete uniformity* for the p-value distribution, as shown for example in Figures 3.4 and 3.12. On the contrary, by Theorem 8 we derive that discrete uniformity implies the weak form of uniformity given by Equation (3.3).

When Equation (3.32) holds, then for ease of terminology we extend to the p-tuple Ω the definition of discrete uniformity.

3.4.1 Number of (discrete uniform) U-valid p-tuples

Hereinafter we maintain the setting defined in §3.3.5, that is, the discrete uniform data distribution case. Under this assumption, given a sample space U of cardinality N_U, it is interesting to compute the number N_{UV} of distinct observable U-valid p-tuples and, among them, the number N_{DU} of those which are also discrete uniform.

As observed in §3.3.4.4, Equation (3.24) provides an upper limit for the number of distinct F-valid p-tuples. The same limit holds for the number of distinct U-valid p-tuples, since in the uniform data distribution scenario by Theorem 6 the two definitions coincide. The actual number of distinct U-valid p-tuples is, however, necessary smaller in virtue of Theorem 5, as we can easily find a couple of subsets $U1, U2$ of U such that $\sum_{u \in U1} f_V(u) = \sum_{u \in U2} f_V(u)$: since we are in the discrete uniform data distribution case, we have $f_V(u) = \frac{1}{N_U}, \forall u \in U$ and, thus, we can simply set $U1 = \{u_1\}$ and $U2 = \{u_2\}$ for two arbitrary distinct samples $u_1, u_2 \in U$.

The number of observable distinct U-valid p-tuples can be computed observing that, because of Theorem 7, all and only the p-tuples defined in Equations (3.30)

have to be taken into account. Hence, we have

$$N_{UV} = \sum_{i=0}^{N_U-1} \binom{N_U - 1}{i} = 2^{N_U-1} \tag{3.33}$$

where i is the number of terms in the p-tuple Ω in Equation (3.30) (with the exception of the final 1) and $\binom{N_U-1}{i}$ is the number of ways they can be chosen. Notice that, in Equation (3.33), $i = 0$ corresponds to the trivial p-tuple $(\omega_1 = 1)$, so the number of meaningful p-tuples is actually $2^{N_U-1} - 1$.

The computation of N_{DU} requires to satisfy both Equation (3.30) (for the definition of U-valid p-tuple) and (3.32) (for the characterization of *discrete uniformity*). Thus we have to find all the tuples of integers $\{k_1, k_2, \ldots, k_{N_\Omega - 1}\}$ such that, for any i,

$$\omega_i = \frac{k_i}{N_U} = \frac{i}{N_\Omega}$$

or, equivalently,

$$k_1 = \frac{N_U}{N_\Omega}$$

observing that, once k_1 is defined, the whole p-tuple is determined by

$$k_i = i \cdot k_1, i = 1, 2, \ldots, N_\Omega \tag{3.34}$$

Since k_1 has to be an integer, the number of solutions is

$$N_{DU} = d(N_U) \tag{3.35}$$

where by $d(N_U)$ we indicate the number of divisors of N_U.

In the specific case (of our main interest) where the sample space size is a power of 2, $N_U = 2^n$, we have

$$N_{UV} = 2^{2^n - 1}$$
$$N_{DU} = n + 1$$

We then note that, among all the observable p-tuples, the number of the discrete uniform ones is negligible.

3.4.2 Number of tests with discrete uniform p-tuple

Of interest is also the number N_{DU}^{T} of tests admitting a discrete uniform p-tuple. This quantity is related to, but different from, the N_{DU} expressed in Equation (3.35), which counts the number of discrete uniform p-tuples. In fact, as we already know from Observation 12, many tests can determine the same p-tuple and, thus, for each discrete uniform p-tuple we have now to compute the number of associated tests.

We observe that for each given k_1 divisor of N_U, the corresponding p-tuple is determined by Equation (3.34) and is made of $\frac{N_U}{k_1}$ terms (compare with Equation (3.30)), thus determining an ordered set partition M_{k_1} made of $\frac{N_U}{k_1}$ subsets of U, with constant probability equal to $\frac{k_1}{N_U}$. Since the underlying data distribution is uniform, it follows that each subset of U contains k_1 elements (refer to Figure 3.6).

So, in order to count the number $N_{M_{k_1}}$ of tests determining the ordered set partition M_{k_1}, we count the number of ways we can partition U in $\frac{N_U}{k_1}$ subsets of k_1 elements each, obtaining

$$\begin{aligned} N_{M_{k_1}} &= \binom{N_U}{k_1}\binom{N_U-k_1}{k_1}\binom{N_U-2k_1}{k_1}\cdots\binom{2k_1}{k_1}\binom{k_1}{k_1} \\ &= \frac{N_U!}{(k_1!)^{\frac{N_U}{k_1}}} \end{aligned} \tag{3.36}$$

Finally, in order to compute N_{DU}^{T}, we still have to sum the right term of Equation (3.36) over the divisors of N_U, obtaining

$$\begin{aligned} N_{DU}^{T}(N_U) &= \sum_{k_1|N_U} N_{M_{k_1}} \\ &= \sum_{k_1|N_U} \frac{N_U!}{(k_1!)^{\frac{N_U}{k_1}}} \end{aligned} \tag{3.37}$$

The first elements of the sequence $(N_{DU}^{T}(N_U), N_U \geq 1)$[21] are:

$1, 3, 7, 31, 121, 831, 5041, 42911, 364561, 3742453, \ldots$

[21] The sequence $(N_{DU}^{T}(N_U), N_U \geq 1)$ is indexed as A061095 in the On-Line Encyclopedia of Integer Sequences (OEIS) [23].

3.4.3 Probability of randomly picking a discrete uniform p-tuple

Now, it is also of interest to evaluate the probability that, uniformly randomly choosing a test, the associated p-tuple is discrete uniform. To this aim, we can compute the ratio $R(N_U)$ of the number of tests with discrete uniform p-tuple, given by Equation (3.37), and the number of possible (non-equivalent) tests, given by Equation (3.24):

$$R(N_U) = \frac{N_{DU}^T(N_U)}{B(N_U)} \tag{3.38}$$

Below we show that $R(N_U)$ tends to 0 as N_U grows to ∞. To do it, we first study the numerator of Equation (3.38), observing that for any $N_U \geq 2$ we have

$$\begin{aligned} N_{DU}^T(N_U) &= \sum_{k|N_U} \frac{N_U!}{(k!)^{\frac{N_U}{k}}} \\ &\leq \sum_{k=1}^{N_U} \frac{N_U!}{(k!)^{\frac{N_U}{k}}} \\ &< \sum_{k=1}^{N_U} \frac{N_U!}{(k!)} \\ &= N_U! \sum_{k=1}^{N_U} \frac{1}{(k!)} \end{aligned} \tag{3.39}$$

Now we analyse the denominator of Equation (3.38), showing that, as N_U goes to ∞, it tends to ∞ as well because of the approximation given by Equation (3.40), see [24]:

$$B(N_U) \approx \frac{N_U!}{2(\ln 2)^{N_U+1}} \tag{3.40}$$

and, hence,

$$\lim_{N_U \to \infty} B(N_U) = \infty$$

L	$N_U = 2^L$	$R(N_U)$
2	4	0.4133
3	8	0.0786
4	16	0.0039
5	32	$1.11 \cdot 10^{-5}$
6	64	$9.00 \cdot 10^{-11}$
7	128	$5.85 \cdot 10^{-21}$
8	256	$2.47 \cdot 10^{-41}$
9	512	$4.41 \cdot 10^{-82}$
10	1024	$1.40 \cdot 10^{-163}$

Table 3.2 Probability of discrete uniform p-tuple

Taking into account Equations (3.39) and (3.40) and applying the well-known Taylor expansion of the Euler's number $e = \sum_{k=0}^{\infty} \frac{1}{k!}$, we obtain

$$\begin{aligned} \lim_{N_U \to \infty} R(N_U) &= \lim_{N_U \to \infty} \frac{N_{DU}^T(N_U)}{B(N_U)} \\ &\leq \lim_{N_U \to \infty} \frac{N_U! \sum_{k=1}^{N_U} \frac{1}{(k!)}}{\frac{N_U!}{2(\ln 2)^{N_U+1}}} \\ &= \lim_{N_U \to \infty} 2(e-1)(\ln 2)^{N_U+1} \\ &= 0 \end{aligned} \tag{3.41}$$

Moreover, directly computing the first values of $R(N_U), N_U = 2^L, L = 2,3,\ldots,10$ (Table 3.2), we see that the probability $R(N_U)$ is extremely low even for very little sample space cardinalities.

Because of Equation (3.41), we can make the following observation, which holds unless the sample space is extremely small and, then, with no practical interest (see Table 3.2):

Observation 13. *Uniformly randomly taking a test among all the possible ones in the discrete uniform data distribution setting, the probability that it determines a discrete uniform p-tuple is negligible.*

This observation will be useful in Section §3.4.4 and later in Section §5.2.2.2, where the strategy to build a (meta)test based on the uniformity (discrete) assumption is examined.

3.4.4 P-value uniformity as meta-test

The uniformity property of the p-values in the continuous case (see Equation (3.2)) can in principle be used to build a meta-test to validate the null hypothesis.

In particular a Goodness-of-Fit test[22] in the *continuous* setting can be implemented based on the following observation: under the null hypothesis, given N observed samples, if we take an arbitrary integer K and split the probability range $[0,1]$ in K disjoint consecutive sub-intervals $I_1, I_2, \ldots, I_K$, each of size $\frac{1}{K}$, then the expected number N_i of samples whose p-value falls in a given sub-interval I_i is constant:

$$N_i = \frac{N}{K}, \forall i \in [1,K] \tag{3.42}$$

The continuous uniformity property of the p-value however does not make sense in the *finite* setting (where we can at most approximate the p-value distribution with a discrete uniform distribution, see Definition 11) and, hence, in general Equation (3.42) does not hold.

In the finite setting, if given the test we are able precisely compute the corresponding p-tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$[23], then we can exactly compute the expected number N_i of samples whose p-value falls in a given sub-interval I_i. If we denote by Ω_i the set of values of Ω belonging to I_i

$$\Omega_i = \{\omega \in \Omega, \omega \in I_i\}$$

then we have

$$N_i = N \sum_{\omega \in \Omega_i} f_{PV}(\omega) \tag{3.43}$$

[22] A Goodness-of-fit test [25], [26], [27] is a hypothesis test which tells how well a statistical model fits the observed data. A typical example is the χ^2 test, see §A.5.

[23] The p-tuple is actually determined by the test specifications, which are of course known to the analyst. This does not however necessarily mean that the p-tuple is easy to derive.

Thanks to Equation (3.6) we can re-write Equation (3.43) as follows

$$N_i = \begin{cases} 0 \text{ if } |\Omega_i| = 0 \\ N(\omega_{M_i} - \omega_{m_i - 1}) \text{ otherwise} \end{cases}$$

where M_i and m_i are the indexes of the maximum and minimum elements of Ω_i:

$$M_i = \max j | \omega_j \in \Omega_i$$
$$m_i = \min j | \omega_j \in \Omega_i$$

and ω_0 is conventionally set to 0 for ease of notation.

Otherwise, if we do not know the resulting p-tuple Ω, we can rely on Equation (3.42) assuming that the uniform distribution is a good approximation. We must however be aware that this is a risky choice, because the actual p-tuple can be quite erratic.

If we consider in particular the uniform data distribution case (see §3.3.5), Observation 13 shows that the p-tuple associated to a generic test is highly unlikely to be discrete uniform. Moreover, Theorem 7 states that any U-valid p-tuple $\Omega = (\omega_1, \omega_2, \ldots, \omega_{N_\Omega})$ (see Definition 10) represents a legitimate p-value distribution (that is, corresponding to a legitimate test) and it is then very easy to build a such distribution not satisfying Equation (3.42), despite the (uniform) null hypothesis being true (thus determining false positive results, increasing the Type I Error probability). To do so we can properly choose

- either a non-uniformly spaced Ω;
- or a big K (relatively to N_Ω).

As an example of the first approach, if we take $\omega_1 > .5$, then all the $\left\lfloor \frac{K}{2} \right\rfloor$ sub-intervals in $\left[0, \frac{1}{2}\right]$ will be empty for any K: $|N_i| = 0, \forall i \leq \left\lfloor \frac{K}{2} \right\rfloor$.

As an example of the second approach, even if we take Ω as uniformly spaced as possible, that is all the p-values are equidistant (and therefore equally likely, see Equation (3.32)), then for any $K > N_\Omega$ some sub-interval will be empty because there are more sub-intervals than observable p-values: $|N_i| = 0$ for at least one $i, 1 \leq i \leq K$. As K grows, more and more sub-intervals will be obviously empty.

In both cases any Goodness-of-Fit test based on the (alleged) p-values uniformity is likely to fail. Based on the above considerations, we can give the following recommendation:

Given a hypothesis test in the discrete setting, let Ω be the set of its p-values, with $\|\Omega\| = N_\Omega$, and let $(0,1]$ be split in K equally sized sub-intervals . Then, in order to mount a Goodness-of-Fit test based on the observed frequencies of the samples in the K sub-intervals, the following constraints should be met: 1. the Ω values are approximately uniformly spaced over (0,1]; 2. K is very small with respect to N_Ω.

Table 3.3 Conditions to build a Goodness-of-Fit test in the discrete setting

Chapter 4

A more general interpretation

In this chapter we propose a more general and abstract interpretation of the concept of hypothesis test, compared to the one given throughout Chapter 2. In particular, in §4.1 some preliminary considerations are developed, enabling the hypothesis test generalized interpretation proposed in §4.2. In §4.3 we leverage the setting developed in §4.2 to reconsider the concepts of test power and Type II Error probability, introduced in §2.1.2, showing how to compute their expected value in the (mostly theoretical) scenario in which we have no hint about the actual structure of the generator under analysis. Later, in §4.4 we formalize the setting of the randomness tests for generators producing uniformly distributed and independent binary sequences, that is, the properties required for cryptographic applications. Relying on the results of §4.3 we also show that, in this model and assuming no information is given about the alternative hypothesis, for a fixed Type I Error probability all the tests are, in a sense, equally effective. Then, in §4.5 we analyse the relation between two generic tests which share the same Type I Error probability, computing the probability of success/failure of a test on a given sample, conditioned on the result of the application of the other test on the same sample. Finally, in §4.6, we analyse some practical issues, showing that only a negligible fraction of the possible tests can be actually implemented and developing some considerations on the concept of test usefulness.

4.1 Preliminaries

In this section, in §4.1.1 we model the extraction process, while in §4.1.2 we formalize the classical interpretation of hypothesis test, in view of the following generalization given in §4.2.

4.1.1 The extraction process

Given a sample space, the null hypothesis can be described by the probability distribution of the extraction process[1], which can be seen as a random experiment. Following the traditional modeling of random experiments, it can be considered as a probability space defined by a triple (U, F, f_V), where:

- U is the sample space, made of all the possible outcomes;
- F is the event space, which in our case can be limited to the set of elementary events, that is the events made of a single outcome. Since we consider only elementary events, U and F are coincident and, hence, hereinafter we will refer to U only;
- f_V is the probability function[2], assigning a probability to each event. In particular, $f_V(u)$ defines the probability that a sample u is randomly extracted from U assuming that the null hypothesis is true.

4.1.2 Classical interpretation of hypothesis tests

In §2.2 two methods are given to test the null hypothesis: the critical value method, see §2.2.1, and the p-value method, see §2.2.2. In the following we refer to the former, since it is essentially equivalent to the latter (see §2.2.3.1) but for our purposes it is more convenient to analyse.

[1] We assume the null hypothesis to be simple, that is, being represented by a unique data distribution (see §2.1). While this choice may appear a bit limiting, it is, in fact, in line with the final target of the chapter, that is, focusing on a single specific distribution, the uniform one, considered in §4.4 and following sections.

[2] The notation f_V is here kept for consistency from Chapter 3. We anticipate that in §4.3 it will be replaced for ease of notation.

In classical terms, given the sample space U and the null hypothesis, a hypothesis test is described by the following elements:

- the definition of a test statistic TS mapping a sample $u \in U$ into a real value $ts = TS(u) \in \mathbb{R}$;
- the definition of a rejection region $RR \subset \mathbb{R}$, such that, if ts falls in the rejection region, then the null hypothesis is rejected, and the definition of the complementary acceptance region $AR \subset \mathbb{R}$, such that, if $ts(u)$ falls in the acceptance region, then the null hypothesis is accepted. The definition of these regions is typically given according to a critical value in the one-tailed models or two critical values in the two-tailed model (see §2.2.1 for the definition of the different models).

Then a test T can be modeled as a composed function:

$$\begin{gathered} T : U \to \mathbb{R} \to OUT \\ T : u \to ts = TS(u) \to \begin{cases} Accept & \text{if } ts \in \text{AR} \\ Reject & \text{if } ts \in \text{RR} \end{cases} \end{gathered} \tag{4.1}$$

where $OUT = \{Accept, Reject\}$ is the space of the possible outcomes of the test, which indicates if observation of u supports or refuses the null hypothesis.

4.2 A generalization of the hypothesis tests

We are now ready to give a more abstract definition of the hypothesis test. Although the usual description, summarized in §4.1.2, makes use of many concepts such as test statistic, critical value(s), acceptance region and rejection region (and, in addition, p-value if we consider the p-value method), here we observe that these are in fact just intermediate tools to reach the ultimate purpose of a hypothesis test, that is simply to determine whether a data sample has to be accepted or rejected, meaning by this that it is consistent or inconsistent with the null hypothesis.

What a hypothesis test basically does is to take in input a data sample and determine if it supports (accepts) or refuses (rejects) the null hypothesis. In this

sense the definition of a generic test given in Equation (4.1) can be re-written as:

$$T : U \to OUT$$
$$T : u \in U \to \{Accept/Reject\}$$

Therefore a hypothesis test determines a partition of the set of the samples into two subsets, one made of the samples mapped to the output $Accept$ and the other one made of the samples mapped to the output $Reject$.

Based on this consideration, we propose the following

Definition 12. *Given a hypothesis test T defined on a data sample space U, the partition of U in two subsets (T_A, T_R) such that*

$$T_A = \{u \in U | T(u) = Accept\}$$

$$T_R = \{u \in U | T(u) = Reject\}$$

is said the essential form *of T.*

Hereinafter, when referring to the essential form of a test T, we will use the terms *acceptance region* and *rejection region* to indicate the two subsets T_A and T_R, respectively. Moreover by *2-partition* we will refer to any partition (of U) in two subsets.

From Definition 12 it is clear that each test, defined in classical terms according to Equation (4.1), admits one and only one essential form. Conversely, any given 2-partition is the essential form of infinitely many tests, as shown by the following theorem.

Theorem 9. *Given a 2-partition (T_A, T_R), there exist infinitely many tests admitting (T_A, T_R) as their essential form.*

Proof. According to the description given in §4.1.2 and in particular to Equation (4.1), a test can be completely defined through the test statistic TS, the acceptance region AR and the rejection region RR. Thus, if we define these components as:

$$TS : ts(u) = \begin{cases} \text{any } a \in AR \text{ if } u \in T_A \\ \text{any } r \in RR \text{ if } u \in T_R \end{cases}$$

where AR and RR are arbitrarily defined, according to the chosen hypothesis test model[3], we obtain infinitely many tests admitting (T_A, T_R) as their essential form. □

Hence, the map that associates a test with its essential form is surjective but not injective: one and only one essential form is associated with a given test, while infinite tests are associated with a given essential form. The reason for this asymmetry lies in the construction of the tests given in Equation (4.1), which require the definition of some (auxiliary) intermediate elements:

- a test statistic TS that sends each sample u into a real value;
- two disjoint and complementary subsets $AR, RR \in \mathbb{R}$.

Given a map $U \to OUT$ there are clearly infinite combinations of TS, AR and RR satisfying the map and which are thus in a sense the same test. Hence we introduce the following

Definition 13. *Two tests are said* indistinguishable *if they have the same essential form.*

We observe that the notion of *equivalence* between tests given by Definition 7 in §3.3.2.1 is stronger than that of *indistinguishability* given by Definition 13, since equivalence requires that, given an arbitrary sample, the two tests associate the same p-value to the sample. This implies that, once the Type I Error value α is fixed, the two samples are eventually mapped on the same output (Accept/Reject) and, hence, that the two tests are indistinguishable by Definition 13. On the other hand, non-equivalent tests (that is, with different mappings from the sample space to the p-value range) can of course be indistinguishable. Again, this asymmetry stems from the fact that the concept of indistinguishability is an abstraction which gets rid of the internal details of the test and considers only the final acceptance/rejection result.

Complementarily to the definition of indistinguishability, we also propose the following

Definition 14. *Two tests admitting different essential forms are said* distinct.

[3]The following definitions can be used: for arbitrary critical values of δ and δ_2, with $\delta < \delta_2$: in the left-tailed model, $RR = (-\infty, \delta]$ and $AR = (\delta, +\infty)$; in the right-tailed model, $AR = (-\infty, \delta)$ and $RR = [\delta, +\infty)$; in the two-tailed model, $AR = (\delta, \delta 2)$ and $RR = (-\infty, \delta] \cup [\delta 2, +\infty)$.

4.3 Test power

As anticipated in Sections §2.1.2 and §2.1.4, given a test, the test power is defined as $1-\beta$, where β is the Type II Error probability, that is, the probability of accepting the null hypothesis when it is concretely false. Maximizing the test power (or, equivalently, minimizing β) is therefore very important. However, in practice, given a test T, it can be very hard to compute β (and thus the power) because the alternative hypothesis is often simply defined as the negation of the null hypothesis and, thus, is made of infinite (alternative) probability distributions.

Nevertheless, in this section we compute the Type II Error probability β in a specific setting, that is, when the alternative distribution is assumed to be *uniformly* taken from the space of all the possible distributions, representing the (mostly theoretical) scenario where the analyst has no clue about the actual distribution.

Let us now consider a generic alternative distribution S, defined as

$$S = \{(u_i, f_V(u_i))\}$$

where u_i is an element of the sample space U and $f_V(u_i)$ is its associated probability. Hereinafter, for ease of notation, we will replace $f_V(u_i)$ with p_i. Moreover, when more convenient, we will equivalently refer to S by the tuple

$$S = (p_1, p_2, \ldots, p_{N_U})$$

where N_U is the cardinality of the set U.

The value of β depends not only on the test $T = (T_A, T_R)$ but on the probability distribution S as well and it is by definition the probability that a sample s from U falls in the acceptance region of T according to the probability distribution S:

$$\beta_T(S) = \sum_{u_i \in T_A} p_i \tag{4.2}$$

We can assume, without loss of generality, that

$$T_A = \{u_{N_U-K+1}, u_{N_U-K+2}, \ldots, u_{N_U}\}$$

for a certain $K, 1 \leq K \leq N_U - 1$. Equation (4.2), thus, can be rewritten as

$$\beta_T(S) = \sum_{j=N_U-K+1}^{N_U} p_j \tag{4.3}$$

Moreover, let Σ be the set of all the possible probability distributions on U. Thus, $\mathbb{E}(\beta_T)$ can be computed as the expected value of $\beta_T(S)$ as S spans over Σ, observing that the contribution of the null distribution, which must be omitted by definition of Type II Error probability, is null since any specific distribution is a single point in the infinite space Σ and, hence, does not impact the overall computation of $\mathbb{E}(\beta_T)$.

Now, let us assume that the alternative distribution is uniformly chosen from the infinite set of possible probability distributions and compute the expected value of β_T. We prove the following theorem.

Theorem 10. *Given a sample space of cardinality N_U, for any test $T = (T_A, T_R)$ on U with acceptance region cardinality $|T_A| = K$, the expected value of the Type II Error β_T, as the alternative probability distribution uniformly varies in the space of all the possible probability distributions, is*

$$\mathbb{E}(\beta_T) = \frac{K}{N_U}$$

Below we present two independent proofs of Theorem 10. The first one is given in §4.3.1 and is based on the calculation of the integral of the probability for a sample to fall in the acceptance region; the second one is provided in §4.3.2 and relies on the action of the symmetric group on the set of the probability distributions. Finally, in §4.3.3, some remarks are reported.

4.3.1 Proof #1

For ease of notation, we first introduce the auxiliary variable ψ_i defined as follows. For $i = 1, 2, \ldots, N_U$, we set

$$\psi_i = \sum_{t=1}^{i} p_t$$

which implies

$$\psi_1 = p_1$$

$$\psi_{i+1} = \psi_i + p_{i+1}, \forall i \in [1, N_U - 1] \tag{4.4}$$

For convenience, we also set $\psi_0 = 0$. Then we prove the two following lemmas:

Lemma 1. *For any pair (i,k), with $k = 0,1,2,\ldots$ and $i = 1,2,\ldots,N_U$,*

$$\int_0^{1-\psi_{i-1}} (1-\psi_i)^k dp_i = \frac{(1-\psi_{i-1})^{k+1}}{k+1}$$

Proof. Because of Equation (4.4),

$$\begin{aligned}
\int_0^{1-\psi_{i-1}} (1-\psi_i)^k dp_i &= \int_0^{1-\psi_{i-1}} (1-\psi_{i-1}-p_i)^k dp_i \\
&= -\left[\frac{(1-\psi_{i-1}-p_i)^{k+1}}{k+1}\right]_0^{1-\psi_{i-1}} \\
&= \frac{(1-\psi_{i-1})^{k+1}}{k+1}
\end{aligned}$$

□

Lemma 2. *For $t \leq N_U$,*

$$\int_0^{1-\psi_{N_U-t}} \cdots \int_0^{1-\psi_{N_U-3}} \int_0^{1-\psi_{N_U-2}} dp_{N_U-1} dp_{N_U-2} \ldots dp_{N_U-t+1} = \frac{(1-\psi_{N_U-t})^{t-1}}{(t-1)!}$$

Proof. Applying Lemma 1 with $k = 3,4,\ldots,t-2$ and $i = N_U - k - 1$,

$$\begin{aligned}
&\int_0^{1-\psi_{N_U-t}} \cdots \int_0^{1-\psi_{N_U-3}} \int_0^{1-\psi_{N_U-2}} dp_{N_U-1} dp_{N_U-2} \ldots dp_{N_U-t+1} \\
&\quad = \int_0^{1-\psi_{N_U-t}} \cdots \int_0^{1-\psi_{N_U-4}} \int_0^{1-\psi_{N_U-3}} (1-\psi_{N_U-2}) dp_{N_U-2} dp_{N_U-3} \ldots dp_{N_U-t+1} \\
&\quad = \frac{1}{2} \int_0^{1-\psi_{N_U-t}} \cdots \int_0^{1-\psi_{N_U-5}} \int_0^{1-\psi_{N_U-4}} (1-\psi_{N_U-3})^2 dp_{N_U-3} dp_{N_U-4} \ldots dp_{N_U-t+1} \\
&\quad = \frac{1}{2}\frac{1}{3} \int_0^{1-\psi_{N_U-t}} \cdots \int_0^{1-\psi_{N_U-6}} \int_0^{1-\psi_{N_U-5}} (1-\psi_{N_U-4})^3 dp_{N_U-4} dp_{N_U-5} \ldots dp_{N_U-t+1} \\
&\quad \vdots \\
&\quad = \frac{(1-\psi_{N_U-t})^{t-1}}{(t-1)!}
\end{aligned}$$

□

We are now ready to give the first proof [4] of Theorem 10.

Proof. We observe that, given an arbitrary probability distribution

$$S = (p_1, p_2, \ldots, p_{N_U})$$

the N_U probabilities p_i can be seen as continuous random variables, uniformly distributed in the $(N_U - 1)$-dimensional domain D defined by the following relations:

$$\begin{cases} 0 \leq p_i \leq 1, i \in [1, N_U] \\ \sum_{i=1}^{N_u} p_i = 1 \end{cases} \tag{4.5}$$

We can now compute the expected value of β_T as the ratio between the sum of the values taken by $\beta_T(S)$ in D, indicated by I_{K,N_U}, and the volume of D, indicated by I_{N_U,N_U}:

$$\mathbb{E}(\beta_T) = \frac{I_{K,N_U}}{I_{N_U,N_U}} \tag{4.6}$$

with

$$I_{N_U,N_U} = \int_0^1 \int_0^{1-p_1} \ldots \int_0^{1-\sum_{i=1}^{N_U-2} p_i} dp_{N_U-1} \ldots dp_2 dp_1 \tag{4.7}$$

$$I_{K,N_U} = \int_0^1 \int_0^{1-p_1} \ldots \int_0^{1-\sum_{i=1}^{N_U-2} p_i} (1 - \sum_{j=1}^{N_U-K} p_j) dp_{N_U-1} \ldots dp_2 dp_1 \tag{4.8}$$

where the integrand in Equation (4.8) derives from Equations (4.3) and (4.5).

In order to compute I_{N_U,N_U} and I_{K,N_U}, Lemmas 1 and 2 turn out to be very useful in explicitly evaluating Equations (4.7) and (4.8). Applying Lemma 2 with $t = N_U$, computation of $I_{(N_U,N_U)}$ is straightforward:

$$\begin{aligned} I_{N_U,N_U} &= \int_0^1 \int_0^{1-p_1} \ldots \int_0^{1-\Psi_{N_U-2}} dp_{N_U-1} \ldots dp_2 dp_1 = \\ &= \frac{1}{(N_U - 1)!} \end{aligned} \tag{4.9}$$

[4] This proof derives from a joint work with Vittorio Bagini and Francesco Stocco [28]; most of the credit of the result goes to them.

On the other hand, computation of I_{K,N_U} is a bit more tricky. Observing that ψ_{N_U-K} depends only on $p_1, p_2, \ldots, p_{N_U-K}$ and, subsequently, applying 2 with $t = K$, we obtain

$$\begin{aligned} I_{K,N_U} &= \int_0^1 \int_0^{1-\psi_1} \ldots \int_0^{1-\psi_{N_U-2}} (1-\psi_{N_U-K}) dp_{N_U-1} \ldots dp_2 dp_1 \\ &= \int_0^1 \ldots \int_0^{1-\psi_{N_U-K-1}} (1-\psi_{N_U-K}) \Big(\int_0^{1-\psi_{N_U-K}} \ldots \int_0^{1-\psi_{N_U-2}} dp_{N_U-1} \ldots dp_{N_U-K+1} \Big) dp_{N_U-K} \ldots dp_1 \\ &= \int_0^1 \int_0^{1-\psi_1} \ldots \int_0^{1-\psi_{N_U-K-1}} (1-\psi_{N_U-K}) \frac{(1-\psi_{N_U-K})^{K-1}}{(K-1)!} dp_{N_U-K} \ldots dp_2 dp_1. \end{aligned}$$

Then, applying Lemma 1 with $k = K, K+1, \ldots, N_U - 1$ and $i = N_U - k$, we conclude

$$\begin{aligned} I_{K,N_U} &= \frac{1}{(K-1)!(K+1)} \int_0^1 \int_0^{1-\psi_1} \ldots \int_0^{1-\psi_{N_U-K-2}} (1-\psi_{N_U-K-1})^{K+1} dp_{N_U-K-1} \ldots dp_2 dp_1 \\ &= \frac{K}{(K+1)!} \int_0^1 \int_0^{1-\psi_1} \ldots \int_0^{1-\psi_{N_U-K-2}} (1-\psi_{N_U-K-1})^{K+1} dp_{N_U-K-1} \ldots dp_2 dp_1 \\ &\vdots \\ &= \frac{K}{(N_U-1)!} \int_0^1 (1-p_1)^{N_U-1} dp_1 = \frac{K}{N_U!} \end{aligned} \tag{4.10}$$

Comparing Equation (4.6) with Equations (4.9) and (4.10), we obtain

$$\mathbb{E}(\beta_T) = \frac{I_{K,N_U}}{I_{N_U,N_U}} = \frac{K}{N_U!}(N_U-1)! = \frac{K}{N_U}$$

thus proving the theorem. □

4.3.2 Proof #2

Here we give the second proof [5] of Theorem 10.

[5] This proof stems from an inspiring talk with Marco Coppola [29].

Proof. Recalling that Σ is defined as the set of all the possible probability distributions on U, we observe that Σ can be partitioned in (infinite) subsets, corresponding to the orbits determined by the action[6] of the symmetric group S_{N_U} on Σ. In particular, given an arbitrary probability distribution

$$S = (p_1, p_2, \ldots, p_{N_U})$$

the resulting orbit $Orb(S)$ is the set of probability distributions which can be reached from S through an element (that is, a permutation) of S_{N_U}, permuting the probabilities associated to the samples. Thus, given a permutation $\pi \in S_{N_U}$, applying π to S we obtain

$$S' = \pi(S) = (p_{\pi(1)}, p_{\pi(2)}, p_{\pi(3)}, \cdots p_{\pi(N_U)})$$

According to Equation (4.3), the Type II Error probability that a sample u, extracted from U according to S', falls in

$$T_A = \{u_{N_U-K+1}, u_{N_U-K+2}, \ldots, u_{N_U}\}$$

is then

$$\beta_T(S') = \sum_{j=N_U-K+1}^{N_U} p_{\pi(u_j)} \tag{4.11}$$

As π spans over the whole set of permutations $S_{N_U} = \{\pi_1, \pi_2, \ldots, \pi_{N_U!}\}$[7], we obtain the orbit of S

$$Orb(S) = \{\pi_1(S), \pi_2(S), \ldots, \pi_{N_U!}(S)\} \tag{4.12}$$

formed by $N_U!$ distributions (possibly with repetitions, if $p_i = p_j$ for some $i, j, 1 \leq i < j \leq N_U$).

Let us denote by $\mathbb{E}(\beta_T^X)$ the expected value of $\beta_T(S')$ as S' spans over a given set X, $X \subseteq \Sigma$. After observing that, under this notation, our final target $\mathbb{E}(\beta_T)$ can be expressed as $\mathbb{E}(\beta_T) = \mathbb{E}(\beta_T^\Sigma)$, from Equations (4.11) and (4.12) we derive the

[6] Given a group G and a set A, a *group action* of G on A is a function $GxA \to A$, $(g,a) \to (g \cdot a)$ such that I) $1_G \cdot a = a, \forall a \in A$; II) $g \cdot (h \cdot a) = (gh) \cdot a, \forall g, h \in G, a \in A$. A such function partitions the set A in equivalence classes, named *orbits*. See [30], pp. 115-119, for an introduction to this topic.

[7] Here the ordering of the permutations π_i with respect to the index i is irrelevant. We simply assume that the set $\{\pi_i\}$, as i spans in $[1, N_U!]$, is composed by all and only the permutations on N_U elements.

expected value of β_T in the orbit of a given permutation S:

$$\mathbb{E}(\beta_T^{Orb(S)}) = \frac{\sum_{i=1}^{N_U!} \left(\sum_{j=N_U-K+1}^{N_U} p_{\pi_i(u_j)} \right)}{N_U!} \tag{4.13}$$

If we rewrite Equation (4.13) as

$$\mathbb{E}(\beta_T^{Orb(S)}) = \frac{\sum_{j=N_U-K+1}^{N_U} \left(\sum_{i=1}^{N_U!} p_{\pi_i(u_j)} \right)}{N_U!} \tag{4.14}$$

we observe that, for any $j \in [N_U - K + 1, N_U]$, the term $p_{\pi(u_j)}$ assumes exactly $(N_U - 1)!$ times each value $p_h, h = 1, \ldots, N_U$, since π_i spans over the whole set of permutations (see §4.3.2.1 for an example). Therefore Equation (4.14) can be rewritten as

$$\begin{aligned} \mathbb{E}(\beta_T^{Orb(S)}) &= \frac{\left(\sum_{j=1}^{K} (N_U - 1)! \right) \left(\sum_{h=1}^{N_U} p_h \right)}{N_U!} \\ &= \frac{\sum_{j=1}^{K} (N_U - 1)!}{N_U!} \\ &= \frac{K \cdot (N_U - 1)!}{N_U!} \\ &= \frac{K}{N_U} \end{aligned} \tag{4.15}$$

Thus, the expected value of the Type II Error probability, as the alternative probability distribution spans in $Orb(S)$, is, for any probability distribution S,

$$\mathbb{E}(\beta_T^{Orb(S)}) = \frac{K}{N_U} \tag{4.16}$$

We observe that the cardinality of the orbits is not constant. More precisely, given $S = (p_1, p_2, \ldots, p_{N_U})$, if all the values p_i are distinct, then in $Orb(S)$ there are $N_U!$ distinct distributions. When, instead, some p_i values collide, the number of distinct resulting distributions consistently decreases, but Equations (4.13), (4.14), (4.15)

and (4.16) still hold (that is, the orbit size is smaller, but the average of the resulting β_T values is unchanged).

Now, in order to extend Equation (4.16) from the orbit of a given (arbitrary) distribution S to the whole set Σ of probability distributions on U, we make the following

Observation 14. *Given an arbitrary number of sets composed of equally-likely real numbers, if the expected value of each set is equal to T (for some constant value T), then the expected value of the union of all the sets is still T.*[8]

Recalling that Σ can be written as the union of infinite disjoint orbits and observing that Equation (4.16) holds for every S, from Observation 14 we derive that

$$\mathbb{E}(\beta_T^{\Sigma}) = \frac{K}{N_U} \tag{4.17}$$

Since $\mathbb{E}(\beta_T) = \mathbb{E}(\beta_T^{\Sigma})$, this proves the theorem. □

4.3.2.1 An example

As a concrete example, let us consider the case $N_U = 4$ and $K = 3$, therefore $U = \{u_1, u_2, u_3, u_4\}$ and $T_A = \{u_2, u_3, u_4\}$. Then, given an arbitrary probability distribution $S = (p_1, p_2, p_3, p_4)$, we have that $Orb(S)$ is made of $N_U! = 4! = 24$ probability distributions $\pi_i, i = 1, 2, \ldots, 24$, listed in Table 4.1 (arbitrarily ordered), where for each permutation π_i we report the probability p'_i associated to each sample u_i ($i = 1, 2, 3, 4$) and the resulting probability to fall in the acceptance region T_A, which, according to Equation (4.3), is equal to $p'_2 + p'_3 + p'_4$. We also highlight that the sum of this probability, as π_i spans over S_4, is $K \cdot (N_U - 1)! = 3 \cdot (3!) = 18$, consistently with Equation (4.15). The resulting Type II Error probability is, hence, exactly $\dfrac{K}{N_U} = \dfrac{3}{4}$, as expected.

$$\mathbb{E}(\beta_T) = \frac{18}{4!} = \frac{18}{24} = \frac{3}{4}$$

[8] The claim can be proven observing that, given two sets A and B, with $A = \{a_1, a_2, \ldots, a_{N_A}\}$ and $B = \{b_1, b_2, \ldots, b_{N_B}\}$, with $\frac{1}{N_A}\sum_{i=1}^{N_A} a_i = \frac{1}{N_B}\sum_{i=1}^{N_B} b_i = T$, then $\sum_{i=1}^{N_A} a_i = N_A T$ and $\sum_{i=1}^{N_B} b_i = N_B T$. Hence, $\sum_{i=1}^{N_A} a_i + \sum_{i=1}^{N_B} b_i = (N_A + N_B)T$ and, finally, $\frac{1}{N_A+N_B}\sum_{i=1}^{N_A} a_i + \sum_{i=1}^{N_B} b_i = T$.

Permutation	**Probability Distribution π_i**				**T_A Probability (β)**
	p'_1	p'_2	p'_3	p'_4	$p'_2+p'_3+p'_4$
π_1	p_1	p_2	p_3	p_4	$p_2+p_3+p_4$
π_2	p_1	p_2	p_4	p_3	$p_2+p_4+p_3$
π_3	p_1	p_3	p_2	p_4	$p_3+p_2+p_4$
π_4	p_1	p_3	p_4	p_2	$p_3+p_4+p_2$
π_5	p_1	p_4	p_2	p_3	$p_4+p_2+p_3$
π_6	p_1	p_4	p_3	p_2	$p_4+p_3+p_2$
π_7	p_2	p_1	p_3	p_4	$p_1+p_3+p_4$
π_8	p_2	p_1	p_4	p_3	$p_1+p_4+p_3$
π_9	p_2	p_3	p_1	p_4	$p_3+p_1+p_4$
π_{10}	p_2	p_3	p_4	p_1	$p_3+p_4+p_1$
π_{11}	p_2	p_4	p_1	p_3	$p_4+p_1+p_3$
π_{12}	p_2	p_4	p_3	p_1	$p_4+p_3+p_1$
π_{13}	p_3	p_1	p_2	p_4	$p_1+p_2+p_4$
π_{14}	p_3	p_1	p_4	p_2	$p_1+p_4+p_2$
π_{15}	p_3	p_2	p_1	p_4	$p_2+p_1+p_4$
π_{16}	p_3	p_2	p_4	p_1	$p_2+p_4+p_1$
π_{17}	p_3	p_4	p_1	p_2	$p_4+p_1+p_2$
π_{18}	p_3	p_4	p_2	p_1	$p_4+p_2+p_1$
π_{19}	p_4	p_1	p_2	p_3	$p_1+p_2+p_3$
π_{20}	p_4	p_1	p_3	p_2	$p_1+p_3+p_2$
π_{21}	p_4	p_2	p_1	p_3	$p_2+p_1+p_3$
π_{22}	p_4	p_2	p_3	p_1	$p_2+p_3+p_1$
π_{23}	p_4	p_3	p_1	p_2	$p_3+p_1+p_2$
π_{24}	p_4	p_3	p_2	p_1	$p_3+p_2+p_1$
Sum of T_A probabilities$= 18(p_1+p_2+p_3+p_4) = 18$					

Table 4.1 Orbit and β values of a generic distribution

4.3.3 Remarks

We have thus proven that, assuming that the (alternative) distributions are uniformly taken from Σ, the expected value for the Type II Error is independent of the specific test (T) and depends only on the cardinalities of the whole sample space and of the acceptance region, taking the form given in Equation (4.17).

We note however that this result is mostly of theoretical interest since the above claim holds if the analyst has no specific information about the actual (alternative) distribution. In practice it can, in fact, happen that the analyst is able to make some hypotheses, for example she/he may suspect some specific bias to be present in the generation process, leading to a non-uniform distribution of the alternative hypotheses and, thus, invalidating the model.

4.4 The Cryptographic Random Test setting

Hereinafter, for the remaining part of Chapter 4, we focus on the specific case of our interest, namely the hypothesis tests to validate generators used for cryptographic applications. As anticipated in §1.3, a strict requirement for these generators is to be indistinguishable from ideal random processes. Hence, in our model we consider random bit generators that produce sequences of a given fixed length L and our null hypothesis is that the sequences are produced uniformly and independently. More formally, the model is defined as follows:

1. The sample space U is made of all the 2^L L-bit possible sequences, thus $N_U = 2^L$;
2. At each sequence generation, the output sequence (extracted from the generator) is uniformly taken from the sample space U, that is, all the 2^L sequences have the same probability 2^{-L} to be produced;
3. Each produced sequence is independent of those produced earlier (and, thus, later).

In this section we analyse some properties holding in the above-mentioned setting, determined by the uniformity property (2) of the extracted sequences (anticipating that the independence property (3), which becomes relevant when we consider a collection of sequences, will be considered in §5.2.2). In particular, in §4.4.1 we analyse the meaning of the Type I Error probability α and in §4.4.2 we study the cardinality of the test space.

4.4.1 The parameter α

Given the sample space U and the null hypothesis of uniformity, let $T = (T_A, T_R)$ be a hypothesis test expressed by its essential form (see Definition 12). Let $K, 0 \leq K \leq 2^L$, be the cardinality of the acceptance region AR, thus also determining the cardinality $(N_U - K)$ of the rejection region RR:

$$|AR| = K, |RR| = 2^L - K \tag{4.18}$$

In general, the probability α of a Type I Error (see §2.1.2) is the probability that $T(u) \in RR$, given a sample u extracted from U according to the data distribution (that is, the probability distribution underlying the null hypothesis). In our specific setting (the null hypothesis being the uniform distribution) α can thus be written as

$$\alpha = \frac{|\{u \in T_R\}|}{2^L} = \frac{2^L - K}{2^L} \tag{4.19}$$

As anticipated in §3.2.2.3, α is not free to take any value in $[0,1]$. More precisely, according to Equation (4.19), the set A of values that α can assume in our setting (as K varies in $[0, 2^L]$) has cardinality $2^L + 1$ and is defined by

$$\mathrm{A} = \{\frac{i}{2^L}, i = 0, 1, \ldots, 2^L\} \tag{4.20}$$

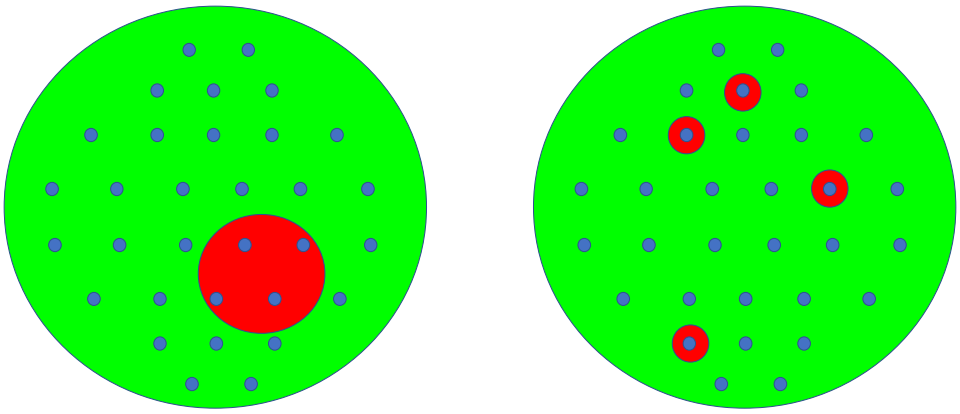
including the two trivial values $\alpha = 0, 1$.

Under the uniform null hypothesis, the meaning of α is also to determine the relation between the cardinality of the acceptance region AR and the cardinality of the rejection region RR. More precisely, by construction (see Equations (4.18) and (4.19)) we have

$$\frac{|AR|}{|RR|} = \frac{1-\alpha}{\alpha} \tag{4.21}$$

A graphical representation is given in Figure 4.1a, with $L = 5, 2^L = 32, \alpha = \frac{1}{8}, |AR| = 28, |RR| = 4$. Each blue circle represents a sample, while the red area (made of 4 samples) is the rejection region and the green area (made of 28 samples) is the acceptance region. We point out that samples in the red area in Figure 4.1a are drawn close to each other for visual simplicity, but this does not reflect any natural structure or concept of distance among the samples. In fact, any other visual arrangement of

the sample space would be correct as well, like for example the unstructured one represented in Figure 4.1b[9].



(a) Structured representation (b) Unstructured representation

Fig. 4.1 Acceptance and rejection regions for a generic test, $\alpha = \frac{1}{8}$

Figure 4.1a (or, equivalently, 4.1b) represents the samples distribution in the acceptance region *AR* and the rejection region *RR*, with the ratio between the cardinalities of the two regions given by Equation (4.21).

4.4.1.1 Test decision on multiple samples

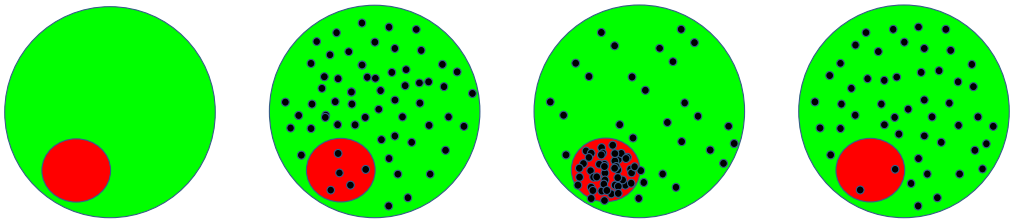
Hence, if we randomly take a set of samples under the null hypothesis, we expect that, observing N samples, the ratio between the number of samples falling in the acceptance region (N_{AR}) and those falling in the rejection region (N_{RR}), with $N = N_{AR} + N_{RR}$, is again

$$\frac{N_{AR}}{N_{RR}} = \frac{1-\alpha}{\alpha} \tag{4.22}$$

This gives a criterion to take a decision on the null hypothesis when dealing with multiple samples, as informally illustrated in Figures 4.2, where black circles represent observed samples and, according to Equation (4.21), the relative area of the rejection region is α (with $\alpha \approx 0.09$ in the figure). Thus, given the rejection region and the acceptance region inside the sample space (Figure 4.2a), according to Equation (4.22) we accept the null hypothesis when the fraction $\frac{N_{RR}}{N}$ of samples falling in the rejection region is approximately α (Figure 4.2b), while we reject the

[9] We observe this is not true for the practical models described in Chapter §2, where closeness and ordering are key concepts. More considerations on this later in §4.6.3.

null hypothesis both when $\frac{N_{RR}}{N}$ is (too) big (Figure 4.2c) or small (Figure 4.2d) with respect to α.



(a) A/R regions (b) Test passed (c) Test failed (I) (d) Test failed (II)

Fig. 4.2 Test decision on the ratio of observed samples in the rejection region

This qualitative description can be made more precise depending on the actual values of the parameters (α and the number of observed samples), as we will do in §5.2.2.1 analysing the testing procedure proposed by NIST.

We emphasize three aspects. First, as anticipated in §2.1.3, the expected fraction of observed samples falling in the rejection region is *exactly* α, not *at most* α. Thus, $\frac{N_{RR}}{N}$ must converge to α as N grows. This implies that the null hypothesis has to be rejected not only in presence of a sample space with too many samples falling in the rejection region, but also when too few samples fall in the rejection region.

Second, the above considerations hold for any value of α; while typical values are 0.01 and 0.05, also smaller or bigger values can be used to build the testing procedure above described. In fact, it makes sense to implement the procedure with different values of α (with values closer to 0.5 easier to manage, because convergence is faster).

Third, acceptance region and rejection region (green and red areas) do not identify *good* or *bad* samples (in the uniform data distribution setting, all the samples are equally likely). They simply partition the set of samples in two subsets, each with an associated probability that a randomly taken observed sample falls inside. Given any test, we can, in fact, swap its acceptance and rejection regions and obtain another valid and consistent test (where *good* and *bad* swap their meaning and the Type I Error probability α is replaced by $1 - \alpha$).

4.4.1.2 Relation between α and β parameters

Finally, let us now consider the relation between the α and β parameters, in view of the analysis of the latter conducted in §4.3, where it is shown that the expected value $\mathbb{E}(\beta_T)$, as the alternative distribution is uniformly taken on the whole set of the possible distributions, is independent of the test and, instead, depends only on the cardinalities of the sample space and the acceptance region (or, equivalently, the rejection region):

$$\mathbb{E}(\beta_T) = \frac{K}{2^L} \tag{4.23}$$

In general, hence, for a given test $T = (T_A, T_R)$ there is no structural relation between α_T and $\mathbb{E}(\beta_T)$, since the former can take any value in the set A (see Equation (4.20)), while the latter has a fixed value determined by Equation (4.23). However, in the uniform data distribution setting, comparing Equations (4.19) and (4.23), we obtain

$$\beta = 1 - \alpha \tag{4.24}$$

independently of the specific test T[10]. Equation (4.24) also tells that, under the uniform data distribution and the assumption that the alternative probability distribution is uniformly taken from the set of all the probability distributions, all the tests with the same α share also the same β and, hence, are in a sense equally effective. This can be summarised in the following

Observation 15. *Under the null hypothesis of uniformity, if no information is given about the alternative hypothesis, then, for a fixed α, no test is* a priori *better than any other.*

4.4.2 Test space cardinality

Each different 2-partition defines a distinct test (see Definition 14), therefore the number of possible distinct tests is equal to the number N_T of distinct 2-partitions of the set U, given by the cardinality of the power set of U, N_U.

[10] We observe that, if we are not under the uniform data distribution, Equation (4.24) can still hold, but only for specific combinations of the rejection region *RR* and the data distribution. More precisely, this happens when the underlying data distribution $\{(u_i, p_i)\}$ is such that $\alpha_T = \sum_{u_i \in RR} p_i = \frac{K}{2^L}$ and, hence, $\alpha_T = 1 - \beta_T$ because of Equation (4.23).

As $|N_U| = 2^L$ we have

$$N_T = 2^{2^L}$$

which includes the two trivial 2-partitions: $(T_A, T_R) = (U, \emptyset)$ and $(T_A, T_R) = (\emptyset, U)$.

Since a value of α is associated to each test, we can count the number N_T^α of different tests with a given α. According to Equation (4.20), α is necessarily in the form $\alpha = \frac{i}{2^L}$ for some integer i in $[0, \ldots, 2^L]$). Under the uniform null hypothesis, N_T^α is equal to the number of 2-partitions (T_A, T_R) for which the rejection region T_R has cardinality $i = 2^L\alpha$ and is therefore

$$N_T^\alpha = \binom{2^L}{i}, i = 2^L\alpha \tag{4.25}$$

With the help of Figure 4.1, we observe that, given a test with Type I Error equal to α, the probability that a sample u, (uniformly) randomly taken from U, falls in the acceptance region or in the rejection region is equal to $1-\alpha$ and α, respectively. On the other hand, given a sample u from U, the probability that it falls in the acceptance region or in the rejection region as a test is (uniformly) randomly taken with with Type I Error equal to α, is equal to $1-\alpha$ and α, respectively. Equivalently, the number $N_{T_A}^\alpha$ of tests for which u falls in the acceptance region and the number $N_{T_R}^\alpha$ for which u falls in the rejection region are

$$N_{T_A}^\alpha = N_T^\alpha(1-\alpha), N_{T_R}^\alpha = N_T^\alpha\alpha$$

respectively.

4.5 Relation between tests

Given the sample space U made of all the possible 2^L binary sequences, $|U| = 2^L$, let $T1$ and $T2$ be two tests defined on U:

$$T1 = (T1_A, T1_R), T2 = (T2_A, T2_R)$$

with

$$\alpha_{T1} = \alpha_{T2} = \alpha$$

We have by construction

$$|T1_A| = |T2_A| = 2^L(1-\alpha)$$

$$|T1_R| = |T2_R| = 2^L\alpha$$

Focusing on the rejection region (complementary considerations hold for the acceptance region), let γ be the ratio of the cardinality of the intersection of $T1_R$ and $T2_R$ over the cardinality of $T1_R$ (or $T2_R$)

$$\gamma = \frac{|T1_R \cap T2_R|}{2^L\alpha} \tag{4.26}$$

In essence γ tells how much $T1_R$ and $T2_R$ overlap, spanning from 0 (the two sets are disjoint) to 1 (the two sets coincide), with any intermediate value indicating a partial overlapping.

The value of γ is constrained by Equation (4.27)

$$max(0, 2-\frac{1}{\alpha}) \leq \gamma \leq 1 \tag{4.27}$$

determined from the obvious condition $0 \leq \gamma \leq 1$ following Equation (4.26) and from the observation that $2^L \geq |T1_R \cup T2_R| = |T1_R| + |T2_R| - |T1_R \cap T2_R| = 2^L\alpha + 2^L\alpha - 2^L\alpha\gamma$ and therefore $1 \geq 2\alpha - \alpha\gamma$ and finally

$$\gamma \geq 2 - \frac{1}{\alpha}$$

From Equation (4.27) we derive the following necessary condition

Observation 16. *In order to have $\gamma = 0$ it is required that $\alpha \leq \frac{1}{2}$.*

Given arbitrary values for α and γ, in the following we analyse the mutual information provided by test $T1$ on test $T2$. In particular, given a sample $u \in U$, we compute the conditional probability that u falls in the acceptance region or in the rejection region of $T2$, assuming first that u falls in the acceptance region of $T1$, then that u falls in the rejection region of $T1$. Probabilities, shown in Equations (4.28) and (4.29), are computed as ratios of set cardinalities according to Figure 4.3,

assuming that all the samples are equally likely and taking into account that

$$T1_A = U \setminus T1_R, T2_A = U \setminus T2_R$$
$$|T1_A| = |T2_A| = (1-\alpha)2^L$$
$$|T1_R| = |T2_R| = \alpha 2^L$$
$$|T1_R \cap T2_R| = \gamma\alpha 2^L$$

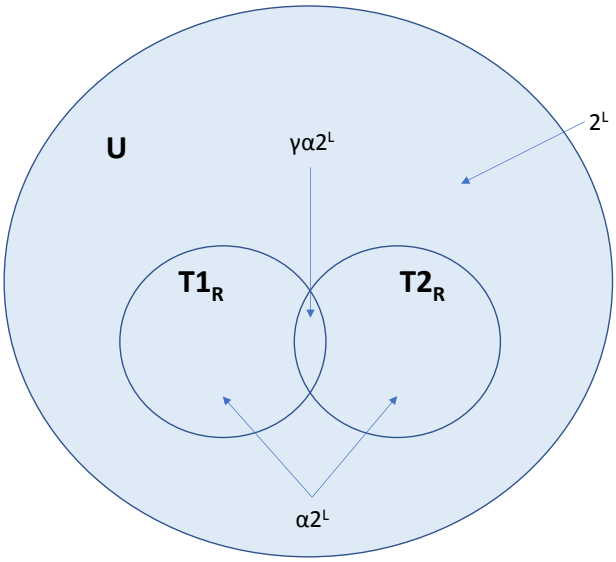


Fig. 4.3 Tests mutual information for a generic γ

The resulting conditional probabilities are thus:

$$T1(u) = \text{Accepted} \Rightarrow T2(u) = \begin{cases} \text{Accepted} & \text{with prob. } \dfrac{1-2\alpha+\gamma\alpha}{1-\alpha} \\ \text{Rejected} & \text{with prob. } \dfrac{\alpha-\gamma\alpha}{1-\alpha} \end{cases} \tag{4.28}$$

$$T1(u) = \text{Rejected} \Rightarrow T2(u) = \begin{cases} \text{Accepted} & \text{with prob. } 1-\gamma \\ \text{Rejected} & \text{with prob. } \gamma \end{cases} \tag{4.29}$$

Notice that probabilities in (4.28) are correctly in the range $[0,1]$ thanks to the constraints in Equation (4.27). The above equations are summarized in Table 4.2.

T1(u)	T2(u)	Prob (T2(u) | T1(u))
Accept	Accept	$\frac{1-2\alpha+\gamma\alpha}{1-\alpha}$
Accept	Reject	$\frac{\alpha-\gamma\alpha}{1-\alpha}$
Reject	Accept	$1-\gamma$
Reject	Reject	γ

Table 4.2 T1 implications on T2

We have two extreme cases, $\gamma = 0$ and $\gamma = 1$, which applying Equations (4.28) and (4.29) result into the following:

- $T1_R$=$T2_R$ ($\gamma = 1$, see Figure 4.4(a)): the rejection regions coincide. This means that test $T1$ and test $T2$ are indistinguishable (see Definition 13) and therefore test $T1$ provides full information on test $T2$. Given $u \in U$ we have

$$T1(u) = \text{Accept} \Rightarrow T2(u) = \text{Accept}$$

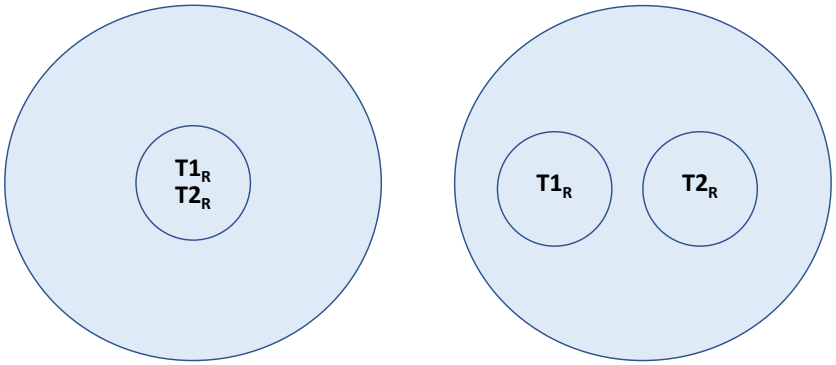
$$T1(u) = \text{Reject} \Rightarrow T2(u) = \text{Reject}$$

- $T1_R \cap T2_R = \emptyset$ ($\gamma = 0$, see Figure 4.4(b)): the rejection regions are disjoint. This means that $T1$ provides partial information on $T2$. Given $u \in U$ we have

$$T1(u) = \text{Accept} \Rightarrow T2(u) = \begin{cases} \text{Accept} & \text{with prob. } \frac{1-2\alpha}{1-\alpha} \\ \text{Reject} & \text{with prob. } \frac{\alpha}{1-\alpha} \end{cases} \tag{4.30}$$

$$T1(u) = \text{Reject} \Rightarrow T2(u) = \text{Accept}$$

We observe that probabilities reported in Equation (4.30) correctly fall in the range $[0,1]$ since $\alpha \leq \frac{1}{2}$ in order to have $\gamma = 0$ (see Observation 16).



(a) $\gamma = 1$ (b) $\gamma = 0$

Fig. 4.4 Tests mutual information for extreme values of γ

The above equations describe the information that a test $T1$ provides about a test $T2$. In particular when we observe the result of $T1$ on a given sample we can gain some information on the expected result of the application of $T2$ on the same sample. When no information is provided from $T1$ about $T2$ we say that the tests are *independent*[11]. By definition of independence, the output of $T2$ is not affected by the output of $T1$. Therefore, remembering that $Pr(T2(u) = Reject) = \alpha$ and $Pr(T2(u) = Accept) = 1 - \alpha$, from the second line of Equation (4.29) it follows that

$$\alpha = \gamma$$

is the (necessary and sufficient) condition to have independence between two tests $T1$ and $T2$, which share the same Type I Error probability α [12].
According to Equation (4.26), it means that, given two tests $T1$ and $T2$ with the same Type I Error probability α, they are independent if and only if α is also the relative size of the intersection of their rejection regions $T1_R$ and $T2_R$ (with respect to each rejection region)

$$\alpha = \frac{|T1_R \cap T2_R|}{2^L \alpha}$$

Finally we note that in the above considerations the roles of $T1$ and $T2$ can be interchanged, leading to the following observation detailed in Table 4.3.

[11] Here we point out that the concept of independence just introduced refers to the relation between tests and is, hence, different from the one presented in §4.4, which is, instead, referred to the relation between sequences.

[12] Alternatively we can solve for α any of the three other following (equivalent) equations derived from Equations (4.28) and (4.29): $1 - \alpha = \frac{1 - 2\alpha + \gamma\alpha}{1 - \alpha}$; $\alpha = \frac{\alpha - \gamma\alpha}{1 - \alpha}$; $1 - \alpha = 1 - \gamma$.

Event	Probability
Pr(T2(u))=Accept | T1(u)=Accept = Pr(T1(u)=Accept | T2(u)=Accept	$\frac{1-2\alpha+\gamma\alpha}{1-\alpha}$
Pr(T2(u))=Reject | T1(u)=Accept = Pr(T1(u))=Reject | T2(u)=Accept	$\frac{\alpha-\gamma\alpha}{1-\alpha}$
Pr(T2(u)=Accept | T1(u)=Reject = Pr(T1(u))=Accept | T2(u)=Reject	$1-\gamma$
Pr(T2(u)=Reject | T1(u)=Reject = Pr(T1(u)=Reject | T2(u)=Reject	γ

Table 4.3 T1 and T2 mutual implications

Observation 17. *Given two tests $T1$ and $T2$ with the same Type I Error probability, information provided by $T1$ on $T2$ is the same as that provided by $T2$ on $T1$.*

We note that Observation 17 is consistent with Observation 15, showing that all the tests with Type I Error probability equal to α are a priori equally effective (assuming no information about the actual analyzed distribution is known in advance).

4.6 Real world tests

Given L and α, with $\alpha = \frac{i}{2^L}$ for some integer i in $[0, 2^L]$, as shown in Equation (4.25) the number of possible different tests is

$$N_T^\alpha = \binom{2^L}{i}$$

which is a huge number for any reasonable values of L and α. Despite all the tests being in a sense equally effective (as noted in Observation 15), in practice only a very small number of them are actually used. It is therefore natural to wonder if tests of practical use can be characterized in some sense. A reasonable answer is that a test of practical use must satisfy two basic requirements:

- it must allow a *compact* definition;

- it must be *easy* to implement.

Although being trivial requirements, they lead to some (mostly qualitative) considerations. Thus, in this section we elaborate a bit on the concepts of *test definition* and *test implementation*, in §4.6.1 and §4.6.2, respectively, and analyze the issues in concretely implementing a generic test, as finally discussed in §4.6.3.

4.6.1 Definition

Given a test T, we say that it admits a *compact* definition if the minimum amount of information required to univocally describe (identify) the test, $I(T)$, is small. In order to quantify $I(T)$ (hereinafter referred to as *definition size*), let us consider the essential form of the test $T = (T_A, T_R)$ and let $I(T_A)$ and $I(T_R)$ be the minimum amount of information required to describe the acceptance region T_A and the rejection region T_R, respectively.

T_R is made of $\alpha 2^L$ samples of U, where each sample u is an L-bit sequence. Since the straightforward description of T_R requires enumerating all its elements in a list $\lambda = \lambda(T_R)$, we can upper limit $I(T_R)$ by the size of the list $\lambda(T_R)$ (expressed in bits)

$$I(T_R) \leq \alpha 2^L L \tag{4.31}$$

The actual value of $I(T_R)$ may be of course (much) smaller than the right side of Equation (4.31). This happens if the list $\lambda(T_R)$ can somehow be lossless compressed. A high lossless compression ratio requires a high redundancy in the structure of T_R, that is, a limited amount of information about $T(R)$ is sufficient to recover its complete definition. However, while the redundancy (and therefore the compressibility factor) can be relevant in some specific cases, for a generic list we do not expect any significant compression. Hence, given a test T, the actual value of $I(T_R)$ is in general not far from $\alpha 2^L L$ and only a negligible fraction of the possible tests admits a small value for $I(T_R)$.

Analogously, in general $I(T_A)$ is not far from $(1-\alpha)2^L L$, with

$$I(T_A) \leq (1-\alpha)2^L L$$

and only a negligible fraction of the possible tests admits a small value for $I(T_A)$.

Since T_A and T_R are complementary in U (T_A=$U \setminus T_R$), once one of the two regions is defined, the other is defined as well and thus

$$I(T) = min(I(T_A), I(T_R))$$

From the above considerations, it follows that, if we take an arbitrary test, we expect that its definition size is high, that is, it requires an intractable amount of information, and that only a negligible fraction of the possible tests admits a small value for their definition size and are, thus, *compact*.

4.6.2 Implementation

When we consider *implementation*, we take into consideration concepts like space, memory and time complexities. For a test to be implementable, it is very important that it

1. can be described by a small code;
2. can be executed in short time;
3. requires little run-time memory.

The above requirements essentially state that the test can be effectively implemented in practice (the exact meaning of *small*, *short* and *little* obviously depends on the specific scenario).

The straightforward way to implement the test is to build a look-up table $[T(u), u \in U]$ as

$$T(u) = \begin{cases} Accept \text{ if } u \notin \lambda(T_R) \\ Reject \text{ if } u \in \lambda(T_R) \end{cases}$$

This approach is in principle always possible, but its implementation (for example a software code) typically requires an intractable amount of space to encode the lookup-up table and, hence, is not a practical option.

The three above-mentioned criteria are often dependent on each other. In particular, we observe that, given a test, a necessary condition to allow a small code is that it admits a compact definition, since any code implementing the test is in fact a definition of the test.

However we also note that a small code size is not enough to guarantee an efficient implementation, since time and memory complexity can still be high, as the following example proves.

4.6.2.1 Example

Let U be the sample space made of the set of the 2^L L-bit sequences and let the test T be defined by

$$T_R = \{u_i \in U, u_i = H(u_{i-1}, i > 0)\} \tag{4.32}$$

where u_0 is a given sample from U and H is a given cryptographic hash function[13] with an L-bit output. In other words T_R is defined as the set of the samples obtained repeatedly applying a hash function starting from a given initial value (since the cardinality of U is finite, at some point the process starts cycling and no new samples are added to the set).

The *definition* of T_R, and thus of the test T, is as simple and compact as Equation (4.32). Since it allows for a direct implementation, the resulting code size is very low as well and the space requirement is fulfilled. However when, given a sample u, we want to verify if $u \in T_R$, we necessarily have to run the whole process defined in Equation (4.32), until we find u or exhaust the look-up table T_R. Or, alternatively, we can do it in an una-tantum pre-processing phase but then we have to store in memory the resulting look-up table T_R. Since the size of T_R is expected to be very large (about $\sqrt{\pi 2^{L-1}}$, see [32], §2.3.1), both options are impracticable and the memory and time requirements are not fulfilled.

4.6.3 Practical tests

From the above sections it is clear that the generic test is not implementable and only a negligible fraction of all the possible tests can be actually of practical use. What we do in practice is then to use tests that are implementable *by construction*. This is exactly what is allowed by the model described in §2.1.1 and followed by virtually every practical test: given a sample from the sample space, a test statistic is

[13] With the definition *cryptographic hash function* we assume that H is computationally non-invertible, that is, given u_i it is practically not possible to recover u_{i-1}. For a formal description see [31], §4.

associated to the sample and then compared to one or two thresholds[14] separating the acceptance region from the rejection region, thus determining if, given a sample, the test succeeds or fails (that is, gives evidence to support or reject the null hypothesis).

With this strategy, coding complexity is thus essentially limited to building the test statistic function and setting the threshold(s). The test statistic function is typically described in an algorithmic way (for example, in a frequency test on random sequences, the test statistic can be defined as "count the number of ones in the given sequence"), thus allowing for a very compact and efficient implementation, and the use of thresholds permits to easily define the acceptance region and the rejection region.

This approach makes tests concretely implementable. The price to pay is that, in doing so, we introduce an artificial ordering among the samples, determined by the associated test statistic real values. This ordering determines a strong correlation between samples with "close" test statistic values, since they result highly likely to provide the same Accept/Reject output. We observe that the concepts of ordering and correlation among samples are not structurally present in the general model described in Sections §4.2 and §4.4.1 and greatly limit the test variability, but allow to overcome the difficulties described in Sections §4.6.1 and §4.6.2 in concretely implementing a generic test.

4.6.3.1 Test usefulness

In practice tests are designed not only to be *easy* to implement but, of course, also to be *useful*. While it seems an obvious requirement, it is not so immediate to define the concept of usefulness since, as shown in §4.4.1.2, all the tests with the same Type I Error probability can be considered in some sense equally effective. The above claim, however, holds only when we have no information about the alternative hypothesis and, thus, we assume that the alternative distribution is uniformly taken from the space of all the probability distributions.

In fact, we can often make some guess about the general structure of the generation process, thus possibly identifying some bias that a properly designed test may be able to detect in the output sequences. A useful observation in this direction is that,

[14]The one-threshold configuration corresponds to the single (left or right) tailed model, while the two-thresholds configuration corresponds to the two-tailed model, see §2.2.1.

while an ideal generation process consists in atomically selecting a sequence among the set of all the sequences of a given length (according to the model adopted in §4.4), what real generators typically do is to produce one short information unit[15] at a time until the desired length is reached. This observation may be the base to build concretely useful tests. For example, when we analyze a generation process producing one bit at a time, it is certainly appropriate to consider the NIST *frequency test* (see for example [17], p. 3-1, §3.1), checking the given sequences for balancing of 0s and 1s. We note, however, that while the frequency test may be effective against a TRNG, relying on a physical process, it is likely ineffective against a PRNG, still producing one bit at a time but based on algebraic properties presenting by construction optimal statistical profile (see §1.3.2). Other tests, like the NIST *Linear Complexity Test* ([17], p 2-24, §2.10), or the one we develop in §6, on the contrary, look for specific structures underlying the generation process and, hence, are expected to be less effective against TRNGs but are more effective against peculiar classes of PRNGs (namely, the Linear Feedback Shift Register (LFSR)s and the LCGs, respectively, as shown in the given references).

In conclusion, when we want to assess the quality of a random generator checking the output sequences against the null hypothesis of uniformity and independence, if we are able to make some assumptions on the generation process and to identify possible intrinsic biases, then we choose tests which are (hopefully) effective in detecting these biases. Hence, the usefulness of a test is not an absolute property but strictly depends on the assumptions we are able to make on the possible biases of the generation process.

[15] The typical information unit is a bit for TRNGs and a byte or a word for (software implemented) PRNGs.

Chapter 5

Statistical tests suites

This chapter introduces a commonly used tool, that is, the *suites* of statistical tests. In §5.1 the need for and the concepts behind this tool are explored and the most common solutions briefly presented. In §5.2 the most widely used suite, namely the NIST-STS provided by NIST, is briefly described and then analyzed in some critical aspects.

5.1 Requirements and common solutions

It is worth observing that the test statistic associated to a given test is a numeric index able to provide a synthesis of the sample data. Of course, precisely because it is a synthesis, it captures only some aspects of the given data.

The class of statistical anomalies that the test can identify, determined by the test statistic, is thus in general not able to detect any other deviations from the null hypothesis. For example, if in the analysis of a random binary sequence the test statistic counts the number of 1s present, it is useful to verify that the sequence respects the balancing criterion (expected number of 0s equal to the expected number of 1s) but in general it does not allow to check other irregularities, such as whether there is a correlation between consecutive bits. Therefore, the effectiveness of the test in providing useful answers depends on the captured properties being relevant or not in the specific use scenario.

In order to be more effective in the analysis, instead of single tests we typically make use of test suites (also known as collections or batteries) made up of several distinct tests with the aim of capturing as many statistical anomalies as possible. Even so, however, no suite of tests is able to identify all the infinite possible anomalies.

The list of tests to be included in a suite must therefore be a tradeoff between implementation efficiency and ability to detect useful anomalies. In principle we try to:

- minimize the number of tests, in order to reduce implementation costs;
- maximize their effectiveness, that is, the ability to capture the non-random properties of interest in the specific use case[1];
- choose independent tests. A subtle point in the definition of a test suite is the independence of the selected tests. Informally, two tests are independent if the output of one of them does not provide any information about the output of the other one. Test independence in a battery is a desirable property for two reasons. First, fresh information about the consistency with the null hypothesis is provided by each test, whereas in case of dependence we would waste computing power to obtain information which is somehow redundant. Second and even more important, independence allows to draw more precise conclusions from multiple tests application.

In literature several suites have been proposed and are commonly used to test the randomness quality of a given generator. The suites differ under many aspects, like the choice of the tests, the implementation efficiency, their flexibility and the user interface. See [35] for an overview and a comparison of the different options. Here we list the most popular batteries:

- Knuth: the first widely used suite was proposed by Donald Knuth ([36]) in 1969. It contains a basic set of 11 tests and is considered the pioneer of systematic randomness testing. However it addresses more specifically

[1] According to the specific use case, an analyst may be more interested in some statistical properties than in others. For example in a Monte Carlo simulation [33], [34] it is vital that the random data used to feed the process appear random, but it is often not required that they are actually random and thus unpredictable. On the contrary, the possibility to repeat an experiment with the same input may result very useful.

real numbers and, therefore, is not so suitable for integer (binary) sequences. Therefore here it is mentioned mostly for historical reasons;

- Diehard: test battery developed by Marsaglia, published in 1995 ([37]). It still represents a useful tool in the statistical community, but is limited on sample size and user-friendliness. An analysis of the battery is proposed in [38].
- Dieharder: an extended version of Diehard, including additional tests, shared by Robert G. Brown ([39]) in 2006. A critical analysis of the suite implementation has been recently proposed in [40].
- TestU01: proposed by L'Ecuyer and Simard [41]. It is an extensive battery, very flexible and with an efficient implementation, first proposed in 2007. A review of the battery can be found in [42];
- PractRand: efficient and versatile suite proposed in 2010 by Chris Doty-Humphrey [43]. It is gaining consideration in the research community. See [44] for a comparison between TestU01 and Practrand;
- gjrand: emerging suite developed by G. Jones [45] contains both a set of PRNGs and a collection of statistical tests;
- NIST-STS: proposed in 2001 by NIST [17] it is the de-facto standard for testing of binary random sequences, with special focus on cryptographic applications. Some analysis of the NIST suite is provided in [46], [47] and [48].

5.2 NIST Statistical Tests Suite

The NIST battery [17] is made of a set of 15 tests (plus some variations for a few of them) which are applied to sequences (samples) of a given arbitrary length. The tests are listed in Table 5.1 along with the statistical property addressed by each of them. Here however we do not analyze in detail the single tests, we just highlight that all of them are built according to the p-value method (§2.2.2) and share the same Type I Error probability $\alpha = 0.01$.

Rather, we comment on some methodological aspects of the suite. Thus, in §5.2.1 we provide some considerations on the definition of null hypothesis used in [17];

Test	**Statistical property analysed**
Frequency (Monobit)	Proportion of zeroes and ones
Frequency within a Block	Proportion of ones within bit blocks
Runs	Total number of runs (uninterrupted sequence of identical bit)
Longest Run of Ones in a Block	Longest run of ones within blocks
Binary Matrix Rank	Rank of disjoint sub-matrices
Discrete Fourier Transform (Spectral)	Peak heights in the Discrete Fourier Transform
Non-overlapping Template Matching	Occurrences of given aperiodic patterns
Overlapping Template Matching	Occurrences of pre-specified target strings
Maurer's Universal Statistical	Number of bits between matching patterns
Linear Complexity	Length of a linear feedback shift register (LFSR)
Serial	Frequency of all possible overlapping patterns
Approximate Entropy	Frequency of repeating patterns in the string
Cumulative Sums (Cusums)	Maximal excursion of the random walk defined by the cumulative sum of adjusted (-1, +1) digits
Random Excursions	Number of cycles having exactly K visits in a cumulative sum random walk
Random Excursions Variant	Total number of times that a particular state is visited (i.e., occurs) in a cumulative sum random walk

Table 5.1 List of tests in the NIST-SP800-22 Rev. 1a suite, [8]

then, in §5.2.2 we elaborate on the methods suggested in [17] to interpret the results obtained by the analysis of more sequences; finally, in §5.2.3 we briefly consider the issue of test independence and in §5.2.4 we comment on the rationale behind choosing (or not) a specific order in tests execution.

5.2.1 On the null hypothesis

In the setting defined in [17] it appears that the object of the null hypothesis is the randomness level of the analyzed sequences. This is explicit for example [17] at pp. 1-3 and 1-4, §1.1.5:

> The null hypothesis under test is that the sequence being tested is random. [...] If a P-value for a test is determined to be equal to 1, then *the sequence appears to have perfect randomness*. A P-value of zero indicates that *the sequence appears to be completely non-random*

or at p. 2-1, §2:

> tests that were developed to test "the randomness of (arbitrarily long) binary sequences"

We do not feel comfortable with this approach which, in our view, is misleading. Instead we believe that the null hypothesis should be expressed in terms of the random generator producing the sequences. More precisely, rather than saying that *the sequence is random* we should define the null hypothesis as *the generator outputs sequences following the uniform distribution and the independence property*. In essence we do not test sequences for randomness, we test generators for uniformity and independence (see §4.4 for the meaning of these properties).

In other words, the focus of the null hypothesis should not be on the properties of the observed sequences (the samples of the hypothesis test model) but on those of the underlying random generator (the source of the samples). In particular, for a given fixed sequence length, under the null hypothesis the random generator is expected to produce all the possible sequences with the same probability and independently of each other.

While this distinction may appear artificial, we believe it captures the essence of the hypothesis test model. In particular we observe that we cannot evaluate the

randomness of an observed sequence, by the very fact that as we observe a sequence, it is completely determined and hence has no randomness at all[2]. Given a sequence (or a set of sequences), the goal of a hypothesis test is to help to determine if the underlying generator is good or not (or, equivalently, if the null hypothesis can be deemed true or false).

If we focus on the generator distribution instead of on the sequences properties, then we recognize that all the sequences are expected to be equally likely and thus there is no concept like *good* or *bad* sequences (in any respect, including their randomness level), as already discussed in §4.4.1.1.

Thus, sequences are not more or less random: they are simply more or less consistent with a given null hypothesis according to a given hypothesis test. It is perfectly acceptable that a sequence passes some tests and fails other tests: what we expect is that the number of passed tests is proportional to $1 - \alpha$ and the number of failed tests is proportional to α, according to the meaning of α as Type I Error probability (assuming the tests are mutually independent). In fact, given a null hypothesis, it is as if each test looks for consistency of observed data with the null hypothesis from a different perspective.

The above distinction seems relevant when considering one of the assumptions made at in [17] at p. 1-5, §1.1.6 with respect to the sequences to be tested:

> Scalability: Any test applicable to a sequence can also be applied to subsequences extracted at random. If a sequence is random, then any such extracted subsequence should also be random. Hence, any extracted subsequence should pass any test for randomness.

Since randomness is not an intrinsic property of a sequence (rather, of the generator), there is no *a priori* guarantee that a subsequence inherits the property of passing a given test[3]. Moreover, referring to the quoted claim, under the null hypothesis of a good generator (that is, producing uniformly distributed and independent sequences),

[2]This could be more formally expressed in terms of entropy (see §A.6), which represents the uncertainty about the sample: if we are able to observe the sample, it has no uncertainty and thus zero entropy.

[3]Of course, a sequence and its subsequences are not independent, thus there may be a correlation between the output of a test applied on the sequence and on any subsequence. However, it is easy to build tests and sequences such that the output of the test applied on a given sequence differs from the output of the same test applied on a certain subsequence. As a trivial example, this happens applying a frequency test (which counts the number of 0s and 1s) on the $2N$-bit sequence built as

we can expect that a given sequence passes not *any* but a fraction $1 - \alpha$ of hypothesis tests[4].

5.2.2 Results interpretation

Due to their probabilistic nature, a very subtle task is to interpret the results provided by the tests. Since a Type I Error probability greater than 0 is present in each test, a strategy to deal with false positives has to be defined. The methodology suggested by NIST ([17] at p. 4-1, §4.1) can be summarised in three steps:

1. fix a length L and collect m sequences (samples) of that length produced by the generator;
2. for each test of the suite execute the test on every sequence and collect the obtained p-values (one for each sequence);
3. take a decision (relative to every specific test) based on the collected p-values.

Table 5.2 NIST strategy for test results interpretation

Let us now analyze the last step, that is, how to decide on a peculiar test (we note that no strategy to decide on the ensemble of the tests is given in [17], in spite of the recommendation to execute multiple tests given in step 2 of the above methodology).

Given the collection of p-values, two criteria are proposed (and below analyzed) in [17], where we believe a strong implicit assumption is made, that is, all the sequences produced and analyzed are independent of each other[5].

$\underbrace{00\ldots0}_{Ntimes}\underbrace{11\ldots1}_{Ntimes}$ (for some N) and then on the first half subsequence $\underbrace{00\ldots0}_{Ntimes}$. It is clear that, for any N, the test is successful on the whole sequence and fails on the subsequence.

[4] In requiring that a fraction $1 - \alpha$ of tests are passed, we implicitly assume that the considered tests are mutually independent. While, strictly speaking, this is not true for the NIST-STS (as later shown in §5.2.3), it does not appear to significantly invalid the above claim.

[5] The mentioned assumption appears quite natural and is, in fact, required by the model we have described in §4.4. However it is not explicitly reported in [17], where the focus is more on the single sequence properties than on the overall generation process.

5.2.2.1 Proportion of successful sequences

Under the null hypothesis (all the sequences are equally likely and independent of each other), if we set the Type I Error probability equal to α and consider m sequences, then the number T of those passing the test follows the binomial distribution[6]

$$T \sim B(m, 1-\alpha) \tag{5.1}$$

since α is by definition the probability to fail the test and then $1-\alpha$ is the probability to pass it. We have obviously $0 \leq T \leq m$. In order to define if the overall test (made of m single tests, one on each sequence) is successful, we can select two integers Min_T, Max_T, with $0 \leq Min_T \leq Max_T \leq m$, and define the acceptance interval as $[Min_T, Max_T]$. If T falls in the interval then the overall test is passed, otherwise it is failed.

Under the null hypothesis, according to Equation (5.1), the probability that T falls in the acceptance interval can be computed as

$$Pr(T \in [Min_T, Max_T]) = \sum_{i=Min_T}^{Max_T} \binom{m}{i} (1-\alpha)^i \alpha^{(m-i)} \tag{5.2}$$

From Equation (5.2) we can immediately derive the Type I Error probability α' of the overall test when the acceptance region is defined by the interval $[Min_T, Max_T]$:

$$\begin{aligned} \alpha' &= 1 - Pr(T \in [Min_T, Max_T]) \\ &= 1 - \sum_{i=Min_T}^{Max_T} \binom{m}{i} (1-\alpha)^i \alpha^{(m-i)} \end{aligned} \tag{5.3}$$

We observe that α' cannot take an arbitrary value in $[0,1]$ but, as a consequence of the discrete nature of the Binomial Distribution, it is forced inside the finite set defined by Equation (5.3) as Min_T and Max_T vary. When however the number of samples m is big enough, the Central Limit Theorem can be used to approximate the discrete

[6] See §A.4.2 for a description of the Binomial Distribution.

Binomial Distribution (Equation (5.1)) with the continuous Normal Distribution[7,8].

$$T \sim N(\mu, \sigma)$$

with μ and σ being the expected value and the standard deviation of T:

$$\begin{aligned} \mu &= m(1-\alpha) \\ \sigma &= \sqrt{m\alpha(1-\alpha)} \end{aligned} \tag{5.4}$$

If we want to consider the proportion of sequences T_P passing the test instead of the absolute value, we just divide by m the expressions in Equation (5.4) and we obtain

$$\begin{aligned} T_P &\sim N(\mu_P, \sigma_P) \\ \mu_P &= 1-\alpha \\ \sigma_P &= \sqrt{\frac{\alpha(1-\alpha)}{m}} \end{aligned} \tag{5.5}$$

In [17], p. 4-2, §4.2.1, the above approximation is used to determine the acceptable proportion of sequences passing the test. More precisely, it is required that

$$T_P \in [\mu_p - 3\sigma_p, \mu_p + 3\sigma_p] \tag{5.6}$$

with μ_P and σ_P given by Equation (5.5).

The definition of the interval in (5.6) corresponds to the well-known 3σ rule for the Normal Distributions [49]. The choice is admittedly arbitrary[9] and determines the following probability $\overline{p}$ that T_P satisfies requirement (5.6):

$$\begin{gathered} \overline{p} = \\ 1 - [F(-3) + (1 - F(3))] = \\ F(3) - F(-3) = \\ 2F(3) - 1 = \\ \approx 0.9973 \end{gathered}$$

[7] See §A.4.2 and §A.4.3 for a description of the Binomial Distribution and the Normal Distribution.

[8] As a rule of thumb, in order for the approximation to hold it is required that $m\alpha \geq 10$ and $m(1-\alpha) \geq 10$, see §A.4.3.1.

[9] From [17], p. 4-2, §4.2.1: "Note that other standard deviation values could be used".

where F is the CDF of the Standard Normal Distribution, exploiting the fact that F is symmetric about 0[10].

In practice $\overline{p}$ is the probability that executing the test on m sequences and calculating the proportion T_P of sequences passing the test, T_P falls in the acceptance interval and thus we consider the overall test passed. Equivalently,

$$\alpha' = 1 - \overline{p} \approx 0.0027 \tag{5.7}$$

is the Type I Error probability associated to the overall test according to the specifications given in [17].

We note however that, even agreeing that the choice of the acceptance interval for T_P is discretionary, there is no apparent specific reason to choose the 3σ setting. We can in fact reverse the process, fixing the desired value for $\overline{p}$ and consistently computing the acceptance interval width. For a generic $\overline{p}$, the acceptance interval can be determined as

$$T_P \in [\mu_p - \gamma\sigma_p, \mu_p + \gamma\sigma_p]$$

with

$$\gamma = F^{-1}(\frac{1+\overline{p}}{2}) \tag{5.8}$$

The value of p and therefore the value of α' (see Equation (5.7)) can be arbitrarily selected, but a natural choice, for the sake of consistency, would be to set $\alpha' = \alpha$ and thus $\overline{p} = 1 - \alpha$.

In the specific case of the NIST-STS we have $\alpha = 0.01$, thus Equation (5.8) would become

$$\gamma = F^{-1}(1 - \frac{\alpha}{2}) = F^{-1}(0.995) \approx 2.5758$$

determining an overall Type I Error probability $\alpha' = \alpha = 0.01$.

5.2.2.2 Uniformity of p-values

A common claim about p-values is that they follow a uniform distribution in $[0,1]$ (see for example [50], p. 2). In Section §3.1 and §3.2, however, we have shown that the p-values uniformity assumption is correct in the continuous case, but is imprecise

[10] Evaluation of F is available in many software applications and in many on-line resources as well.

in the discrete setting. Nevertheless, a second criterion suggested by NIST in [17] (p. 4-3, §4.2.2) to evaluate a collection of a given test on m sequences is to compute the p-value for each sequence and then check that the m resulting values are uniformly distributed. Since the considered setting is discrete, we think that the proposal is somewhat questionable, as argued below.

The above-mentioned procedure proposed by NIST consists in dividing the interval $(0, 1]$ in ten equal sub-intervals, counting the number of p-values falling in each sub-interval and then determining if they can be deemed uniformly distributed. For this purpose two methods are proposed: a (qualitative) visual check of the resulting histogram and a (quantitative) application of a χ^2 test (see §A.5). We here focus on the second method, as it can be treated in a more precise way.

The procedure (hereinafter referred to as *NIST procedure* for simplicity) is described in more detail in Table 5.3 for a generic number of sub-intervals K (with $K = 10$ for the specific case of the NIST procedure[11]), emphasizing that the NIST procedure is itself a hypothesis test, where the null hypothesis is that the p-values are uniformly distributed. Looking at Table 5.3, we observe that in step 1 the p-value is computed according to the underlying test definition; in step 2 the extreme value 0 is excluded because we know from Observation 4 that all the observable p-values are greater than 0; in step 4 the expected value (η) for each h_i is computed according to Equation (3.42), that is, under the uniform null hypothesis (that the p-values are uniformly distributed); finally in step 5 the resulting χ^2 index is checked against the suggested Type I Error probability $\alpha'' = 0.0001$ in order to determine if the set of sequences supports or rejects the null hypothesis.

[11]The procedure, defined by a fixed value $K = 10$ by NIST in [17] (p. 4-3, §4.2.2), has been here generalized to an arbitrary value of K, because later two distinct values ok K will be actually considered in our analysis, namely $K = 10$ and $K = 100$.

Given an integer K and m independent sequences $\{s_t, t = 1,2 \ldots m\}$,

1. for each sequence s_t, compute the corresponding p-value pv_t

$$pv_t = PV(s_t)$$

2. divide the $(0,1]$ interval in K equal sub-intervals

$$I_i = \left(\frac{i-1}{K}, \frac{i}{K} \right], i = 1,2,\ldots,K$$

3. compute the number h_i of p-values falling in I_i

$$h_i = \#\{pv_t \in I_i, t = 1,2,\ldots m\}, i = 1,2,\ldots,K$$

4. compute the χ^2 index with $K-1$ degrees of freedom

$$C = \sum_{i=1}^{K} \frac{(h_i - \eta)^2}{\eta} \text{ with } \eta = \frac{m}{K}$$

5. compute the p-value of C according to the χ^2 distribution

$$pv_C = PV(C)$$

6. output *Accept* if $pv_C > \alpha''$, with $\alpha'' = 0.0001$; output *Reject* otherwise.

Table 5.3 NIST procedure to check p-values uniformity

Now we observe that, since the χ^2 index is itself a p-value, if the approach followed by NIST were correct, then we would expect that executing the procedure M times (that is, taking $M \cdot m$ independent random sequences, grouping them in M blocks, each one made of m sequences, and running the procedure for each block), the M resulting χ^2 values were uniformly distributed as well. Hence, we propose the procedure defined in Table 5.4, hereinafter referred to as *meta-procedure*, which follows the principle discussed in §3.4.4.

Given $M \cdot m$ independent sequences

1. group them in M blocks $B_i, i = 1,2,\ldots,M$, each made of m sequences;
2. for each block B_t, compute the χ^2 index c_t applying the NIST procedure of Table 5.3 (steps 5 and 6 are here ignored);
3. divide the $(0,1]$ interval in 10 equal sub-intervals
$$\overline{I}_i = \left(\frac{i-1}{10}, \frac{i}{10}\right], i = 1,2,\ldots,10$$
4. compute the number $\overline{h_i}$ of p-values falling in $\overline{I}_i$
$$\overline{h_i} = \#\{c_t \in \overline{I}_i, t = 1,2,\ldots M\}, i = 1,2,\ldots,10$$
5. compute the χ^2 index with 9 degrees of freedom
$$\overline{C} = \sum_{i=1}^{10} \frac{(h_i - \overline{\eta})^2}{\overline{\eta}} \text{ with } \overline{\eta} = \frac{M}{10}$$
6. compute the p-value of $\overline{C}$ according to the χ^2 distribution
$$\overline{pv_C} = PV(\overline{C})$$
7. output *Accept* if $\overline{pv_C} > \overline{\alpha''}$, with $\overline{\alpha''} = 0.01$; output *Reject* otherwise.

Table 5.4 Meta-procedure to check p-values uniformity

We notice that steps 3-6 of the meta-procedure are basically the same as steps 2-5 of the NIST procedure, replacing the χ^2 indexes computed on the (m) sequences with those computed on the (M) blocks, K with 10, m with M and finally $\alpha'' = 0.0001$ with (the arbitrary but more standard) $\overline{\alpha''} = 0.01$. We also note that the chosen number of intervals (10) is arbitrary, but in Table 5.4 it has been fixed to make the description easier to read, also in view of the fact that 10 is the actual value used in the following analysis. We remark that the number of intervals in the meta-procedure is 10, irrespective of the value ok K in the NIST procedure.

Finally, in order to report the results of the meta-procedure applied on the NIST-STS battery ([17]), we remark that the set of the meta-procedure parameters (which include those of the NIST procedure) is composed of

- m, the number of sequences;
- M, the number of blocks, each made of m sequences;
- K, the number of sub-intervals of $(0,1]$ considered in the NIST-procedure;
- L, the bit length of the sequences.

In an ongoing joint work with Alessandro Giacchetto, following his Master's thesis [51], we have implemented the meta-procedure defined in Table 5.4 for all the tests proposed in [17] (see Table 5.1). We have done it in two configurations, first with $K = 10$ sub-intervals, as suggested in [17] (p. 4-3, §4.2.2), with the other parameters set to $M = 100$ blocks and $m = 1000$ sequences; then with $K = 100$, maintaining the same values as before for $M = 100$ and $m = 1000$[12].

The sequences used for the experiments are $L = 1,000,000$-bit long and were generated using the cryptographic primitive AES-GCM (see §A.7), which is firmly believed by the cryptographic community to be a high quality pseudo-random generation mechanism, statistically indistinguishable from ideal random generators, and able to produce (statistically) independent sequences. More on this and the validation methodology in §6.5.1.

Results found vary greatly with the specific test considered. For example, if we consider the Approximate Entropy Test, the resulting p-values distribution looks approximately uniform in both configurations ($K = 10$ and $K = 100$), as shown in Figures 5.1 and 5.2, obtaining convincing χ^2 and corresponding p-values[13]: $(\overline{C_{10}} = 5.6, \overline{pv_{C-10}} = 0.78)$ and $(\overline{C_{100}} = 6.2, \overline{pv_{C-100}} = 0.72)$, respectively.

[12]The two configurations reported explain the above-mentioned choice to have 10 sub-intervals in the meta-procedure (Table 5.4). As explained in §A.5, the value of η (step 5 of the meta-procedure), that is, the expected number of p-values falling in each sub-interval, must be at least 5 (and preferably more). Since $M = 100$ for both configurations, choosing 10 sub-intervals leads to $\eta = \frac{100}{10} = 10$, which is a safe value for the application of the χ^2 test.

[13]See steps 5 and 6 of the meta-procedure described in Table 5.4.

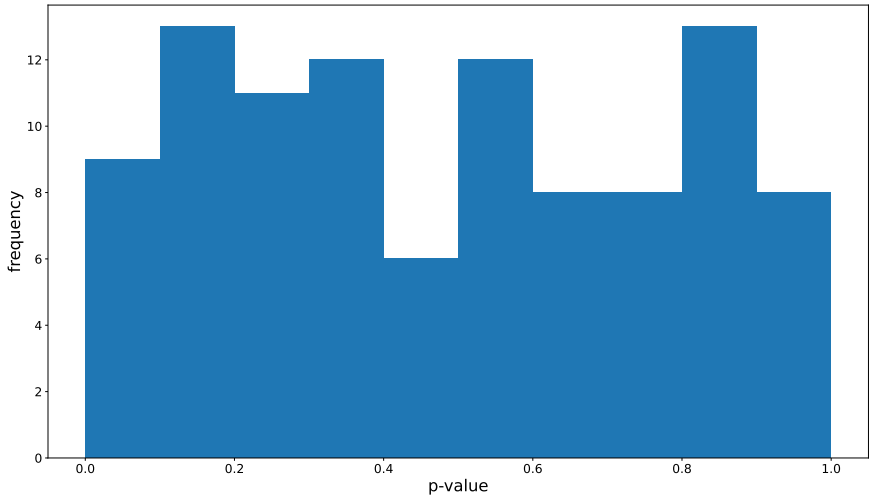


Fig. 5.1 Approximate Entropy Test, χ^2 p-values distribution, $K = 10, \overline{pv_{C-10}} = 0.78$

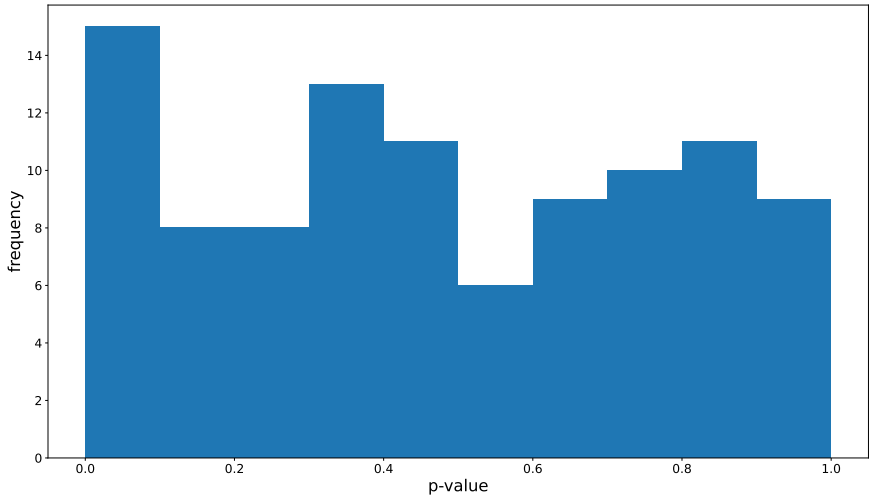


Fig. 5.2 Approximate Entropy Test, χ^2 p-values distribution, $K = 100, \overline{pv_{C-100}} = 0.72$

However, other tests do not exhibit an equally good behaviour, the Discrete Fourier Transform Test and the Binary Matrix Rank Test being the worst ones. In particular, if we consider the Discrete Fourier Transform Test, the resulting p-values are well distributed for $K = 10$ ($\overline{C_{10}} = 11.80, \overline{pv_{C-10}} = .22$) as shown in Figure 5.3,

but fail miserably for $K = 100$ ($\overline{C_{100}} = 880.20, \overline{pv_{C-100}} = 10^{-183}$) as shown in Figure 5.4.

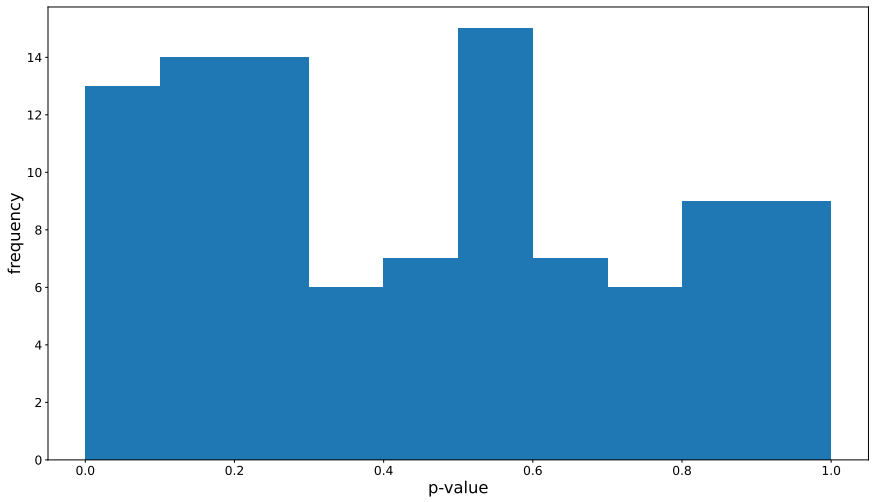


Fig. 5.3 Discrete Fourier Transform Test, χ^2 p-values distribution, $K = 10, \overline{pv_{C-10}} = 0.22$

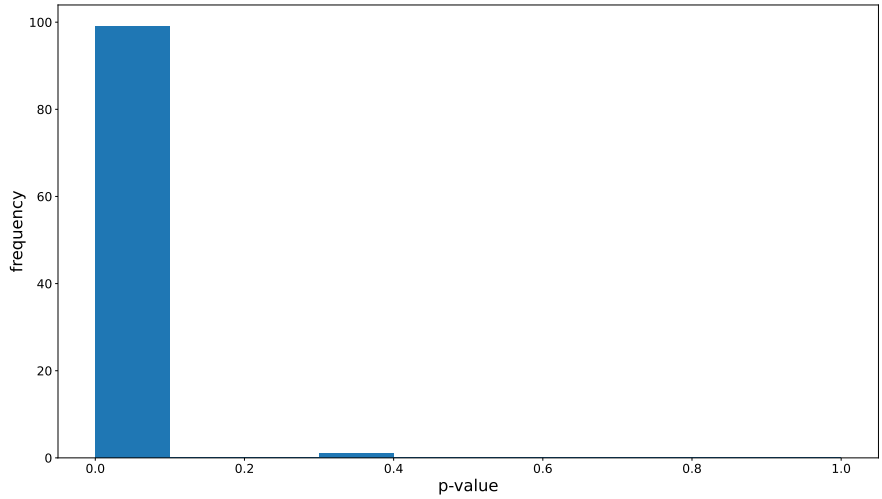


Fig. 5.4 Discrete Fourier Transform Test, χ^2 p-values distribution, $K = 100, \overline{pv_{C-100}} = 10^{-183}$

Similarly, the Binary Matrix Rank Test behaves well ($\overline{C_{10}} = 5.80, \overline{pv_{C-10}} = 0.76$) for $K = 10$ and fails for $K = 100$ ($\overline{C_{100}} = 480, \overline{pv_{C-100}} = 1.1 \cdot 10^{-97}$) as shown in Figures 5.5 and 5.6, respectively.

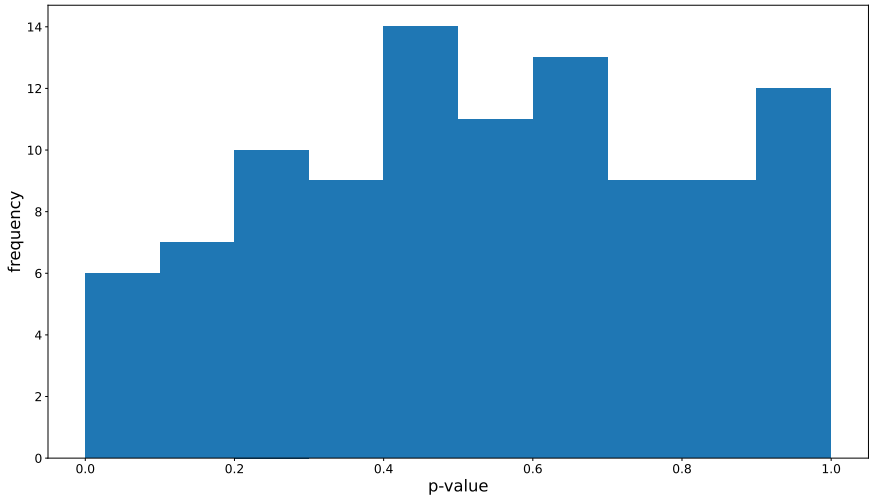


Fig. 5.5 Binary Matrix Rank Test, χ^2 p-values distribution, $K = 10, \overline{pv_{C-10}} = 0.76$

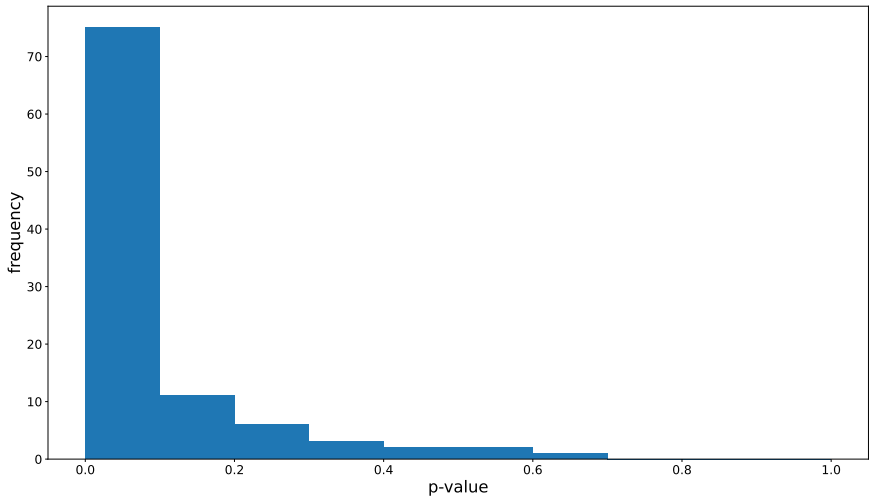


Fig. 5.6 Binary Matrix Rank Test, χ^2 p-values distribution, $K = 100, \overline{pv_{C-100}} = 1.1 \cdot 10^{-97}$

In [17], §4.2.2, no justification for the NIST procedure (Table 5.3) is given, apart from the assumption that p-values are uniformly distributed. However the above-mentioned assumption on the p-values uniformity, while being correct in the continuous case, is just an approximation in the finite discrete case, as discussed in §3.4, and in fact it is not clear what NIST exactly means, when in [17] at p. 4.3, §4.2.2, they write

> The distribution of P-values is examined to ensure uniformity.

A plausible interpretation is that by *uniformity* they mean *discrete uniformity* (see Definition 11). We remark however, as pointed out in Observation 13, that discrete uniformity, although possible, is extremely unlikely for a generic test, that is, for an arbitrarily test taken among all the possible ones in the given setting[14], where the sample space is made of all the possible binary sequences of a given length and the null hypothesis is that they are (independent and) uniformly distributed

Nevertheless, as argued in §3.4.4, the NIST procedure can still be useful in the discrete case if the p-tuple associated to the test (see §3.2) is actually approximately (discrete) uniform and the number of intervals (K, in Table 5.3) is small with respect to the number of distinct possible p-values. The tricky point is that both conditions require to know the p-tuple associated to the test (that is, the set of observable p-values, §3.2.1), which, although completely determined by the test definition, is not always easy to derive analytically. If the p-tuple is not known, then using the discrete uniform approximation can be a risky choice, because the above-mentioned conditions may not be met.

In order to understand why different tests have so different behaviours, as shown in Table 5.5, we have analysed through simulations the p-tuples associated to the NIST-STS tests presented in [17]. Though necessarily imprecise, this method allows to derive some useful information. In particular Table 5.5 reports for each test the number of distinct p-values observed executing the test on $100,000$ random sequences[15] . We note a great variability in the results. Not surprisingly, the Binary Matrix Rank Test and especially the Discrete Fourier Transform Test are among

[14]Of course no test is *randomly taken*, since it always has a rationale behind its design. However, if discrete uniformity is not considered as part of the requirements, then, with regard to this specific property, the resulting test can be a priori considered randomly taken.

[15]When more variants of a given test are defined, we report the average value of the number of distinct p-values observed.

Test	**Number of distinct observed p-values**
Frequency (Monobit)	1,658
Frequency within a Block	18,759
Runs	91,052
Longest Run of Ones in a Block	92,153
Binary Matrix Rank	4,112
Discrete Fourier Transform (Spectral)	419
Non-overlapping Template Matching	16,804
Overlapping Template Matching	94,818
Maurer's Universal Statistical	95,067
Linear Complexity	93,955
Serial	21,199
Approximate Entropy	95,205
Cumulative Sums (Cusums)	2,953
Random Excursions	59,484
Random Excursions Variant	52,538

Table 5.5 Number of distinct observed p-values on $100,000$ test executions

those with only a very small number of different observed p-values (it is of course possible that, increasing the number of tested sequences, some new p-value appears, but it is quite likely that only a small increment could be observed)[16].

Given the low number of distinct p-values observed for these tests, it is natural to check how the $100,000$ p-values are distributed in $(0,1]$. The result is shown in Figures 5.7 and 5.8, respectively.

[16]Frequency (Monobit) Test and Cumulative sums (Cusums) Test have a low number of observed p-values as well. However, here we focus on the Binary Matrix Rank Test and the Discrete Fourier Transform Test because these are the ones exhibiting the worst profiles in terms of $\overline{pv_C}$ values.

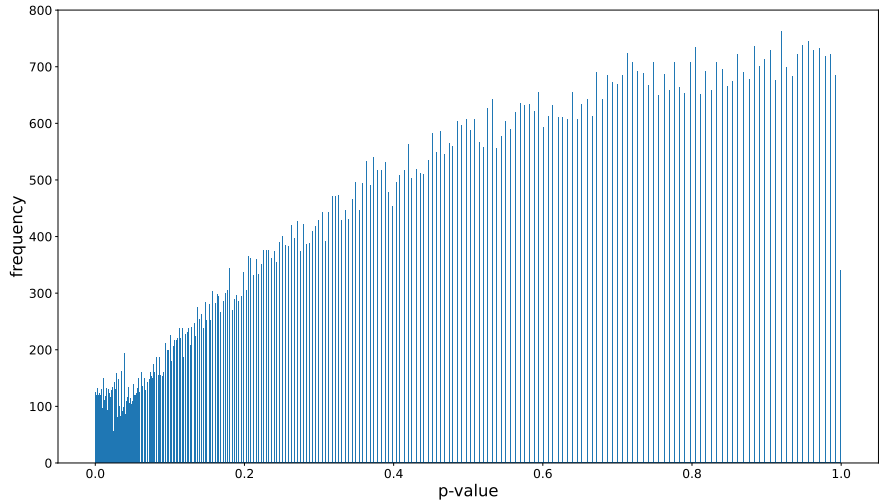


Fig. 5.7 Discrete Fourier Transform Test, distribution of p-values

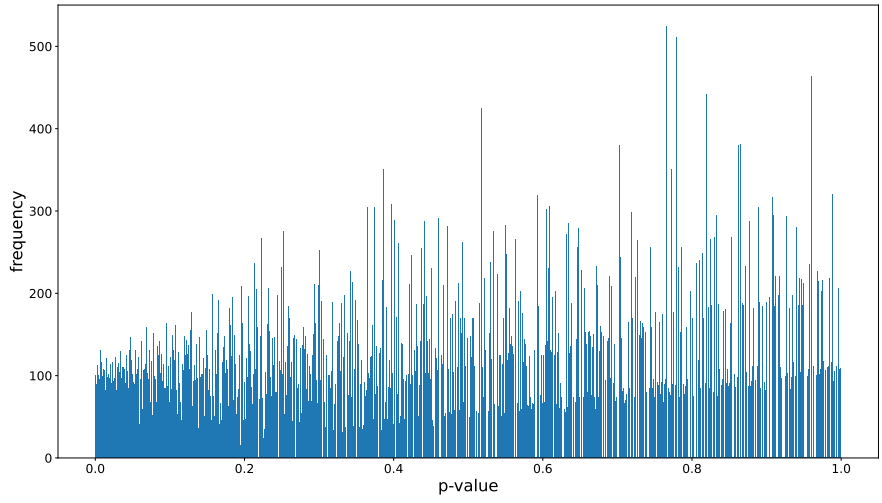


Fig. 5.8 Binary Matrix Rank Test, distribution of p-values

It clearly appears that the resulting distributions for both tests (and especially for the Discrete Fourier Transform Test) are far from satisfying the discrete uniformity property (requiring all the values to be equally spaced and equally likely, see Theo-

rem 8). On the contrary, as shown in Figure 5.9, the situation for the Approximate Entropy Test is much better, though not optimal[17].

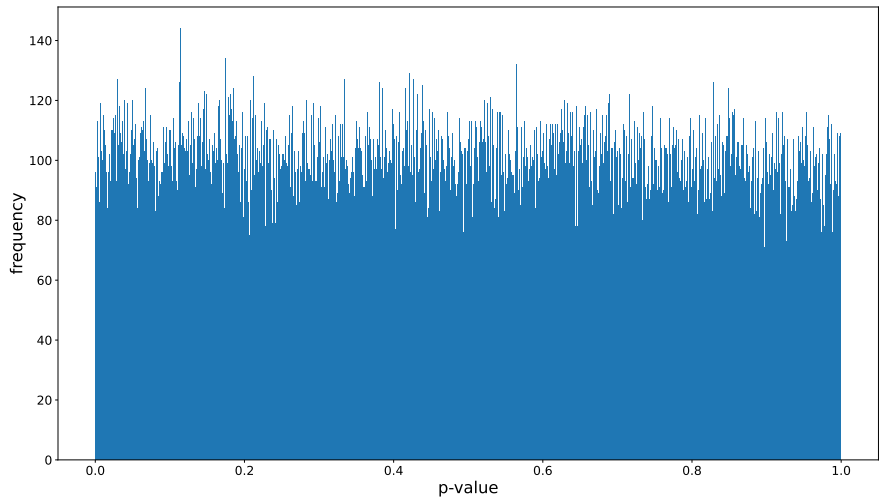


Fig. 5.9 Approximate Entropy Test, distribution of p-values

According to the conclusions of §3.4.4, which require the p-tuple to be (approximately) discretely uniform, this explains why the Discrete Fourier Transform Test and the Binary Matrix Rank Test grossly fail the meta-procedure with $K = 100$.

Moreover in §3.4.4 it is also required that the number of sub-intervals (that is, K in the NIST procedure reported in Table 5.3) is small with respect to the cardinality of the set of p-values associated to the considered test. This looks consistent with the observation that all the NIST-STS test pass the χ^2 meta-procedure with $K = 10$ but some of them fail with $K = 100$. Although no explicit rationale is given in [17] behind the choice to set $K = 10$ in the NIST-STS procedure, it is reasonable that a so small value was chosen to mitigate the discretization effect, as discussed in §3.4.4, and make the NIST procedure (Table 5.3) likely to be passed under the null hypothesis of p-values uniformity.

We also note that the Type I Error probability α'' suggested in the procedure is as small as 0.0001, which is clearly inconsistent with the values $\alpha = 0.01$ associated

[17]The χ^2 value on the observed p-values for the Approximate Entropy Test is equal to 0.0069 if computed with $1,000$ bins and to 0.0945 if computed with $10,000$ bins, while it is $< 10^{-300}$ in both cases for the Binary Matrix Rank Test and the Discrete Fourier Transform Test.

to each single test and $\alpha' = 0.0027$ (see Equation (5.7)) associated to the Proportion of Sequences Passing a Test described in [17], §4.2.1. We believe that this small value for α'' has been chosen to increase the success rate for sequences produced under the null hypothesis. In fact, our simulations have shown that using a more standard value for α'' would cause an unacceptably high rejection ratio for χ^2 values produced according to the NIST procedure under the null hypothesis.

For example if we set $\alpha'' = 0.01$ and we zoom in Figures 5.4 and 5.6 on the $[0, 0.01]$ range, as shown in Figures 5.10 and 5.11, we observe that 95 out of 100 p-values for the Discrete Fourier Transform Test and 35 out of 100 p-values for the Binary Matrix Rank Test are smaller than 0.01, thus determining an abnormally high failure rate, much bigger than the expected $\alpha'' = 0.01$.

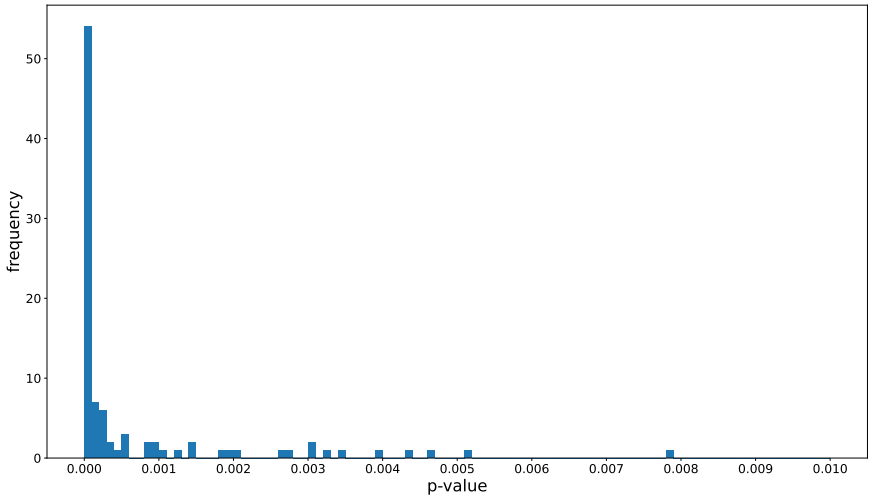


Fig. 5.10 Discrete Fourier Transform Test, χ^2 p-values distribution, K=100, range $[0, 0.01]$

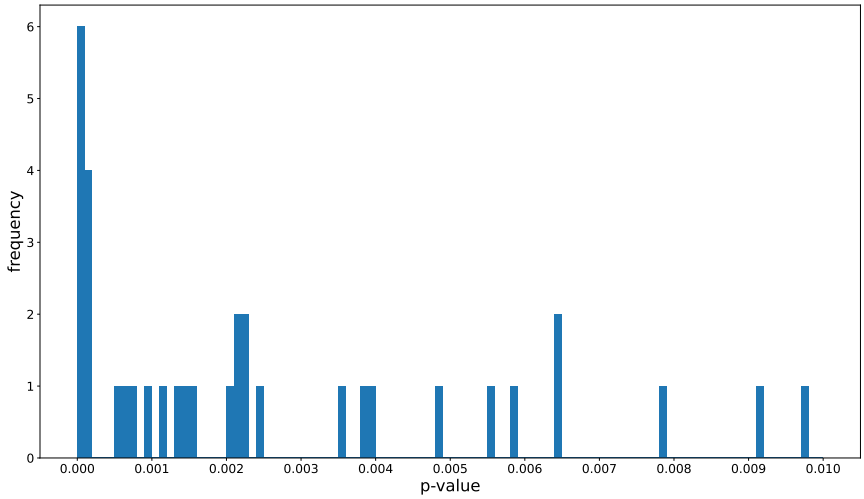


Fig. 5.11 Binary Matrix Rank Test, χ^2 p-values distribution, K=100, range $[0, 0.01]$

Moving the threshold α'' to 0.0001 obviously increases the success rate, but only limitedly. According to our simulations, the number of p-values below α'' are in fact reduced to 38 and 6 (out of 100) for the Discrete Fourier Transform Test and the Binary Matrix Rank Test, respectively, still corresponding however to unacceptably high failure rate (again very far from the expected 0.0001). We emphasize that this inconsistency is not due to poor randomness quality of the generators producing the analyzed sequences, but instead to an imprecise test model (based on the wrong assumption of p-value uniformity in the discrete case).

We conclude this section with the following observation.

Observation 18. *As the p-values get denser, they also become less likely.*

This effect, which is implied by Equation (3.6) and is especially evident in Figure 5.7, is somehow consistent with the NIST-STS procedure, reported in Table 5.3. In other words, the expected uniformity in the distribution of the frequencies of the p-values observed in each of the K sub-intervals is in some sense correct (with easy to find exceptions, as above shown). However this result does not come from the deemed p-value uniformity (which is not guaranteed moving from the continuous setting to the discrete finite one), but instead from Observation 18.

5.2.3 Test independence

As anticipated in §5.1, collections made of mutually independent tests are preferable because they are easier to characterize. Essentially, given two independent tests and a sample to analyse, the probability that both tests are passed, both are rejected or one is passed and the other one is rejected, is easy to compute from the individual probabilities, while the case of dependence can be much harder to deal with.

In [17] at p. 4.4, §4.6, some analysis is reported on the redundancy (and therefore dependency) of the tests in the NIST-STS, concluding that the

> The degree of duplication among the tests seems to be very small.

Despite this, a few papers in literature [52], [53], [54], including the recent Master's thesis by Alessandro Giacchetto that we have supervised [51], show that some tests in the NIST-STS have a statistically significant mutual correlation. It is however unclear if and how this can concretely impact on the global reliability of the suite.

5.2.4 Test execution order

In [17], p. 2-1, §2, it is suggested that the Frequency Test is run before the others

> Since this supplies the most basic evidence for the existence of non-randomness in a sequence, specifically, non-uniformity. If this test fails, the likelihood of other tests failing is high.

and later, in [17], p. 3-1, §3.1, the hint is reinforced

> All subsequent tests are conditioned on having passed this first basic test

In Observation 17 we have shown, however, that two tests with the same Type I Error probability (as is in the case of the NIST-STS, where for all the tests the parameter α is set to 0.01), are symmetric in the information they provide on each other, that is, the probability of failure of the frequency test given that another test fails is the same as the probability of failure of the second test given that the frequency test fails. In this sense, uniformity of bits in a given sequence, despite being the most

intuitive property, is in fact no more important than other properties[18]. Moreover, as emphasized in §5.2.1, according to our null hypothesis, we do not look for bit uniformity inside a given sequence, instead we look for sequence uniformity (and independence) among sequences produced by the generator extraction process.

A more convincing reason to run the frequency test first is somehow given in the same [17],p. 2-1, §2, where emphasis is put on the computational effort required by the tests

(The most time-consuming statistical test is the Linear Complexity test)

implicitly pointing out that it is advantageous to run the frequency test first because it is faster. We note, however, that, even if the frequency test fails, a good practice is to run the other tests as well, in order to minimize the risk of rejecting a valid null hypothesis (Type I Error on the ensemble of the tests) on the basis of a single test failure. In this view, the order of execution of the tests looks, hence, irrelevant.

[18]For example, a pseudo-random sequence generated according to a linear recurring sequence is, in general, well balanced (that is, passes the frequency test) but fails the Linear Complexity Test, which is part of the NIST-STS (see [17], p. 2-24, §2.10).

Chapter 6

A new hypothesis test suite

In this chapter we present a simple (toy) collection of hypothesis tests, named DECT Suite[1]. Although it was primarily designed to provide practical hands-on experience with the concepts discussed in previous chapters, as we will see, the suite proves to be quite effective in certain concrete situations.

The chapter is organized as follows. After some necessary preliminaries in §6.1, in §6.2 we introduce an intermediate hypothesis test, named K-Test, whose purpose is to make the the subsequent section smoother to treat. Moreover, the analysis of the K-Test leads to an insightful view on the practical difference between exact and approximate calculations (specifically the Binomial Distribution and its Normal Distribution approximation) in terms of the specific application scenario. Then, in §6.3, we define two further hypothesis tests, one named DECT-W Test and based on the K-Test, the other named DECT-Q Test. Both of them have multiple instances, which hereinafter are collectively referred to as DECT Suite. In the next section, §6.4, we describe the software implementation of the DECT-W Test and the DECT-Q Test, which is also fundamental in their validation procedure[2] reported in detail in §6.5 and leading to a good confidence that the two tests are correctly defined and implemented (and, thus, the DECT Suite as well). Later, in §6.6, we report on the application of the DECT Suite to detect statistical biases in a widely used class of PRNGs, namely the LCGs. Somewhat unexpectedly, it appears that our toy suite performs better than the most commonly used collections of statistical tests when it comes to this specific

[1] DECT stands for "Decimation Test".

[2] As later discussed in §6.5.1, when defining a test, a strong validation procedure is of paramount importance.

class of generators. Finally, in §6.7, we propose a list of topics, discussed in this chapter, which seem worthy of further investigation.

6.1 Preliminaries

Let L be an integer power of 2, $L = 2^l$, and let λ be an integer, $\lambda < l$. Given an L-bit sequence $S = (s_1, s_2, \ldots, s_L)$, let us split S in N adjacent disjoint 2^λ-bit-long blocks, with $N = 2^n$ and

$$n = l - \lambda. \tag{6.1}$$

Hereinafter we assume that

$$n \geq 5, \lambda \geq 7 \tag{6.2}$$

and, hence, $l \geq 12$ (the rationale behind this choice is given in §6.3.3). Let us indicate the i-th block ($i = 1, 2, \ldots, N$) with $D_i = (d_{i,j}, j = 1, 2, \ldots, 2^\lambda)$, where $d_{i,j} = s_{(i-1)2^\lambda + j}$, and, for each $j = 1, 2, \ldots, N$, we define

$$K_j = \sum_{i=1}^{N} d_{i,j} \tag{6.3}$$

Thus, as summarized in Figure 6.1, K_j counts how many times $d_{i,j}$ is equal to 1 as we move across all the N blocks $D_i, i = 1, 2, \ldots N$. Differently said, K_j sums all the bits of the sequence obtained decimating S by a factor 2^λ with offset j.

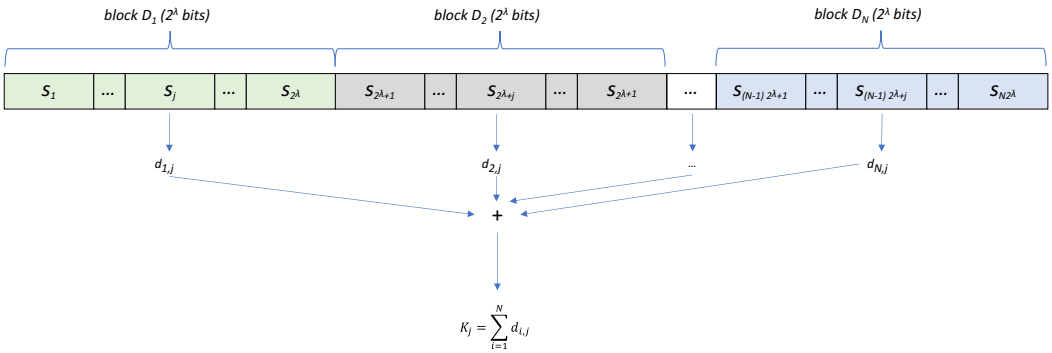


Fig. 6.1 From sequence to K_j

Under the null hypothesis that the sequences analyzed are produced according to the uniform distribution, we expect that, for any j, K_j follows the Binomial

Distribution

$$K_j \sim B\left(N, \frac{1}{2}\right) \tag{6.4}$$

with mean value μ_N and standard deviation σ_N equal to

$$\mu_N = \frac{N}{2}, \sigma_N = \frac{\sqrt{N}}{2} \tag{6.5}$$

respectively (see §A.4.2).

Alternatively, the well-known Normal Distribution approximation to the Binomial Distribution (see §A.4.3.1) provides an approximate distribution of K_j as

$$K_j \dot{\sim} \mathcal{N}(\mu_N, \sigma_N^2) \tag{6.6}$$

From Equation (6.6), applying the standard normal transformation (see §A.4.3.2)

$$z_j = \frac{K_j - \mu_N}{\sigma_N} \tag{6.7}$$

we move to the easier to manage Normal Standard Distribution $\mathcal{N}(0,1)$

$$z_j \dot{\sim} \mathcal{N}(0,1) \tag{6.8}$$

6.2 K-Test

Here we define an intermediate test, referred to as K-Test, upon which in §6.3.1 we will build the DECT-W Test. The K-Test is a hypothesis test in the two-tailed model (see §2.2), taking in input an integer value in $[0,N]$ (that is, the range of the K_j values). The null hypothesis is that the input variable is distributed according to Equation (6.4) and we would like to define the rejection region of the K-Test so that the associated Type I Error probability α is 0.1[3]. However, from Definition 3 we know that only some values of α are admissible. Thus, in general, the best we can

[3]We remark that the choice of the model (*two-tailed*) and of the α value (0.1) are completely arbitrary, with the only goal to split the sample space in two complementary sets (acceptance and rejection regions) in order to (indirectly) verify the uniform null hypothesis, in line with the methodology presented in §4.4.1.1. Other models and other values of α would work as well, simply resulting in different tests.

do is to choose a rejection region whose corresponding Type I Error probability is as close as possible to the target value 0.1.

The section is organized as follows: in §6.2.1 we provide an exact definition of the acceptance and rejection regions, up to the point where it is computationally feasible. Then in §6.2.2 we go further, relying on approximations, thus arriving at the complete definition of the acceptance and rejection regions. Then, in §6.2.4 we compare the exact and approximated computation methods and provide some related remarks.

6.2.1 Acceptance and rejection regions definition

Given $N = 2^n$, for an integer n, in the (symmetric) two-tailed model the acceptance and rejection regions are in the form

$$AR_N = [\frac{N}{2} - v_N + 1, \frac{N}{2} + v_N - 1]$$

$$RR_N = [0, \frac{N}{2} - v_N] \cup [\frac{N}{2} + v_N, N]$$

for an integer value $v_N, 1 \leq v_N \leq \frac{N}{2}$. The Type I Error probability associated to the couple (N, v_N) can be precisely computed as

$$\alpha_{v_N}^N = 1 - \frac{1}{2^N} \sum_{i=\frac{N}{2}-v_N+1}^{\frac{N}{2}+v_N-1} \binom{N}{i} \tag{6.9}$$

Thus, given N, we choose v_N such that the resulting $\alpha_{v_N}^N$ is the closest possible value to 0.1, as reported in Table 6.1, which is made of 4 columns: the first one containing the value of $n = \log_2 N$, the second one indicating the chosen value of v_N, the third one showing the resulting acceptance region AR_N, and finally the fourth one giving the corresponding Type I Error probability $\alpha_{v_N}^N$. For any n, the rejection region RR_N is obviously equal to $[0,N] \setminus AR_N$[4].

[4]In Table 6.1 and later in Table 6.2, for ease of reading only the acceptance regions are reported, with the rejection regions easily determined simply complementing the acceptance regions with respect to the whole set $[0,N]$.

n ($N = 2^n$)	v_N	AR_N ($RR_N = [0,N] \setminus AR_N$)	$\alpha^N_{v_N}$
5	5	$[12,20]$	0.11018417
6	7	$[26,38]$	0.10342188
7	10	$[55,73]$	0.09269003
8	14	$[115,141]$	0.09131209
9	19	$[238,274]$	0.10191701
10	27	$[486,538]$	0.09762363
11	38	$[987,1061]$	0.09743877
12	53	$[1996,2100]$	0.10086355
13	75	$[4022,4170]$	0.09970991
14	106	$[8087,8297]$	0.09925997
15	149	$[16236,16532]$	0.10085643
16	211	$[32558,32978]$	0.10006578
17	298	$[65239,65833]$	0.10028519
18	422	$[130651,131493]$	0.09966389
19	596	$[261549,262739]$	0.10000026
20	843	$[523446,525130]$	0.09986509
21	1192	$[1047385,1049767]$	0.09985797
22	1685	$[2095468,2098836]$	0.09996577
23	2383	$[4191922,4196686]$	0.09992915
24	3369	$[8385240,8391976]$	0.10001614

Table 6.1 Acceptance and rejection regions and Type I Error probability

6.2.2 Extension to arbitrary values of N

As N grows, exact computation of the Type I Error probability (see Equation (6.9)) becomes very time-consuming and, eventually, unfeasible. Consequently, Table 6.1 reports values only for N up to 2^{24}. For higher values of N we can, however, rely on the Normal Distribution approximation to the Binomial Distribution and on its standardization, expressed in Equations (6.6), (6.7) and (6.8). This allows to work, more comfortably (though approximately), on the Normal Standard Distribution $\mathscr{N}(0,1)$, whose CDF is hereinafter referred to as $F_{\mathscr{N}(0,1)}$. Moreover we indicate

with τ the value taken by its inverse in 0.95

$$\tau = F^{-1}_{\mathcal{N}(0,1)}(0.95) \approx 1.64485362$$

which also imply, due to the symmetry of $\mathcal{N}(0,1)$ about the origin,

$$-\tau = F^{-1}_{\mathcal{N}(0,1)}(0.05)$$

Thus, if we set

$$RR' = (-\infty, -\tau] \cup [\tau, +\infty) \tag{6.10}$$

as the rejection region in the standard setting, then the resulting Type I Error probability is $.05 + .05 = 0.1$.

Now we can move back to the original value of N, derive the rejection region RR'_N and compare it with the rejection region RR_N reported in Table 6.1 (obtained by exact computation), in order to understand how good is the approximation found in Equation (6.10). To do so, we simply apply the inverse of Equation (6.7) to the extremes of RR', setting $z_j = \pm\tau$:

$$RR'_N = (-\infty, \mu_N - \tau\sigma_N] \cup [\mu_N + \tau\sigma_N, +\infty)$$

with σ_N and μ_N defined according to Equation (6.5). Complementarily, we have

$$AR'_N = (\mu_N - \tau\sigma_N, \mu_N + \tau\sigma_N) \tag{6.11}$$

In Table 6.2 the first column reports the value of n, while the second column shows the corresponding acceptance region AR'_N, as for Equation (6.11). Finally, the third column contains the acceptance region with integer extremes

$$\begin{aligned} AR''_N &= [\lceil \inf(AR'_N) \rceil, \lfloor \sup(AR'_N) \rfloor] \\ &= [\lceil \mu_N - \tau\sigma_N \rceil, \lfloor \mu_N + \tau\sigma_N \rfloor] \end{aligned} \tag{6.12}$$

obtained from the corresponding AR'_N; we observe that AR'_N and AR''_N turn out to be equivalent since the input to the K-Test is an integer value.

Considering the values of N for which exact computation AR_N is available (that is $N \leq 2^{24}$) and comparing the values of AR_N in Table 6.1 and AR''_N in Table 6.2, we observe that the acceptance regions (and, thus, the rejection regions) determined

n ($N = 2^n$)	AR'_N ($RR'_N = [0,N] \setminus AR'_N$)	AR''_N ($RR''_N = [0,N] \setminus AR''_N$)
5	(11.348, 20.652)	[12, 20]
6	(25.421, 38.579)	[26, 38]
7	(54.695, 73.305)	[55, 73]
8	(114.841, 141.159)	[115, 141]
9	(237.391, 274.609)	[238, 274]
10	(485.682, 538.318)	[486, 538]
11	(986.781, 1061.219)	[987, 1061]
12	(1995.365, 2100.635)	[1996, 2100]
13	(4021.562, 4170.438)	[4022, 4170]
14	(8086.729, 8297.271)	[8087, 8297]
15	(16235.125, 16532.875)	[16236, 16532]
16	(32557.459, 32978.541)	[32558, 32978]
17	(65238.250, 65833.750)	[65239, 65833]
18	(130650.917, 131493.083)	[130651, 131493]
19	(261548.499, 262739.501)	[261549, 262739]
20	(523445.835, 525130.165)	[523446, 525130]
21	(1047384.999, 1049767.001)	[1047385, 1049767]
22	(2095467.670, 2098836.330)	[2095468, 2098836]
23	(4191921.998, 4196686.002)	[4191922, 4196686]
24	(8385239.340, 8391976.660)	[8385240, 8391976]
25	(16772451.995, 16781980.005)	[16772452, 16781980]
26	(33547694.680, 33561169.320)	[33547695, 33561169]
27	(67099335.990, 67118392.010)	[67099336, 67118392]
28	(134204253.359, 134231202.641)	[134204254, 134231202]
29	(268416399.980, 268454512.020)	[268416400, 268454512]
30	(536843962.718, 536897861.282)	[536843963, 536897861]
31	(1073703711.960, 1073779936.040)	[1073703712, 1073779936]
32	(2147429749.436, 2147537546.564)	[2147429750, 2147537546]
33	(4294891071.920, 4295043520.080)	[4294891072, 4295043520]
34	(8589826794.873, 8590042389.127)	[8589826795, 8590042389]
35	(17179716731.841, 17180021628.159)	[17179716732, 17180021628]
36	(34359522775.745, 34359953964.255)	[34359522776, 34359953964]
37	(68719171843.681, 68719781636.319)	[68719171844, 68719781636]
38	(137438522311.491, 137439384688.509)	[137438522312, 137439384688]
39	(274877297107.363, 274878516692.637)	[274877297108, 274878516692]
40	(549754951522.982, 549756676277.018)	[549754951523, 549756676277]
...	...	...

Table 6.2 Estimated acceptance region by Normal Distribution approximation

by the two methods (namely, the exact computation and the Normal Distribution approximation) are equivalent:

$$\text{for } N \leq 2^{24}, AR_N \equiv AR''_N, RR_N \equiv RR''_N$$

We also observe from Table 6.1 that, as N increases, the difference between $\alpha^N_{\nu_N}$ and the target probability 0.1 tends to decrease (although not monotonically). Hence, we can safely extend Table 6.1 to arbitrary values of N using the Normal Distribution approximation to define the acceptance region (as done in Table 6.2 up to $N = 2^{40}$, but easily extensible to higher values, if needed) and setting $\alpha^N_{\nu_N} = 0.1$.

6.2.3 K-Test definition

We can now define the K-Test as follows. Given N and an integer k, $0 \leq k \leq N$,

$$K - Test(k) = \begin{cases} \text{Passed if } k \in AR_N \\ \text{Failed if } k \in RR_N \end{cases} \tag{6.13}$$

where AR_N and RR_N are complementary regions defined as in Table 6.1 if $N \leq 2^{24}$ and as for Equation (6.12) and Table 6.2 if $N > 2^{24}$. Consistently, for any $N \leq 2^{24}$ the Type I Error probability is

$$Pr(k \in RR_N) = \alpha^N_{\nu_N}$$

according to Table 6.1, while it can be assumed to be equal to 0.1 for any $N > 2^{24}$. For example, if $N = 14$, we have $AR_N = [8087, 8297]$, $RR_N = [0, 8086] \cup [8298, 16384]$ and $\alpha^N_{\nu_N} = 0.09925997$.

6.2.4 Methods comparison

Above we have described two methods to compute the rejection region, namely the exact computation (§6.2.1) and the Normal Distribution approximation (§6.2.2). Then we have verified that the two methods produce the same rejection region, for any $N \leq 2^{24}$.

However, here we observe that relying on the Normal Distribution approximation allows for very fast computation and, moreover, permits to obtain values for arbitrarily large values of N (reported up to $N \leq 2^{40}$ in Table 6.2). These considerations make the Normal Distribution approximation generally preferable. Nevertheless, a caveat is necessary, given in the following

Observation 19. *With exact computation, the actual Type I Error probability is precisely known. On the contrary, with the Normal Distribution approximation it can only be (in general imprecisely) assumed to be as determined by the model considered.*

Whether this discrepancy is relevant or not depends on the specific case. In general, the more we use a wrong approximation, the more the effect on the final correctness is amplified. The following (qualitative) observation can then be useful (as for example later in §6.3.3):

Observation 20. *If the difference between the actual and the assumed Type I Error is small, the impact on a single application of the test may be negligible. However, if we mount a (meta)test based on multiple samples analysis (as discussed in §4.4.1.1), then the impact of the difference on the overall result may be meaningful.*

In our specific setting, we observe from Table 6.1 that the disagreement between the correct Type I Error probability $\alpha^N_{\nu_N}$ and the estimated value 0.1 can vary significantly with N, see for example the extreme opposite cases $N = 5$ and $N = 19$.

6.3 DECT Suite

In this section we introduce two tests, namely the DECT-W Test and the DECT-Q Test, which are defined and commented in §6.3.1 and §6.3.2, respectively. Then, in §6.3.3, we give some remarks common to both.

Both tests take the set $K = \{K_j, j = 1 \dots 2^\lambda\}$ (see Equation (6.3)), compute an overall index on K and then associate a p-value to that index. In particular, the DECT-W Test focuses on the average profile of K, while the DECT-Q Test analyses its extreme behaviour. The following description of both tests assumes that n (with

1. Compute K_j, for $j = 1, 2, \ldots, 2^\lambda$, according to Equation (6.3);
2. apply the K-Test on K_j, for $j = 1, 2, \ldots, 2^\lambda$, according to Equation (6.13), see §6.2;
3. compute W as the number of K_j values failing the K-Test;
4. set $\alpha = \alpha^N_{v_N}$, as defined in Table 6.1, if $n \leq 24$; $\alpha = 0.1$ otherwise;

   set $\mu_W = 2^\lambda \alpha$;

   set $\sigma_W = \sqrt{2^\lambda \alpha(1-\alpha)}$;

   compute $z_W = \dfrac{W - \mu_W}{\sigma_W}$;
5. compute $p_W = 2 \cdot (1 - F_{\mathcal{N}(0,1)}(|z_W|))$.

Table 6.3 DECT-W Test

$N = 2^n$) and λ are given, with $n \geq 5, \lambda \geq 7$ (see Equation (6.2)). Differently said, for a given sequence of length $L = 2^l$ (with $l \geq 12, l = n + \lambda$), we can consider any pair (λ, n) such that

$$\begin{aligned} \lambda &= 7, 8, \ldots, l-5 \\ n &= l - \lambda \end{aligned} \tag{6.14}$$

The pair of values (λ, n), given according to Equation (6.14), defines a specific instance for both tests.

6.3.1 DECT-W Test

The definition of the DECT-W Test test is given by table 6.3. Steps 1, 2 and 3 are self-explanatory and end with the computation of W as the number of K_j values failing the K-Test. We observe that, under the null hypothesis of uniformity, W follows the Binomial Distribution with 2^λ samples and success probability equal to α and, hence, can be approximated by the Normal Distribution with parameters

1. Compute K_j, for $j = 1, 2, \ldots, 2^\lambda$, according to Equation (6.3);
2. set $\mu_N = \dfrac{N}{2}$;

   set $\sigma_N = \dfrac{\sqrt{N}}{2}$;

   compute $z_j = \dfrac{K_j - \mu_N}{\sigma_N}$, for $j = 1, 2, \ldots, 2^\lambda$;
3. compute $Q = \max_{j \in [1, 2^\lambda]} (|z_j|)$;
4. set $\mu_Q = F^{-1}_{\mathcal{N}(0,1)}(1 - \frac{1}{2^\lambda})$;

   set $\beta_Q = F^{-1}_{\mathcal{N}(0,1)}(1 - \frac{1}{e2^\lambda}) - \mu_Q$;

   set $\gamma_Q = e^{-e^{-\frac{Q-\mu_Q}{\beta_Q}}}$;

   compute $p_Q = 1 - \gamma_Q^2$.

Table 6.4 DECT-Q Test

(μ_W, σ_W^2), see §A.4.2 and §A.4.3.1:

$$W \sim B\left(2^\lambda, \alpha\right) \dot{\sim} \mathcal{N}(\mu_W, \sigma_W^2) \tag{6.15}$$

Step 4 applies the normal standard transformation to W, obtaining z_W which is, hence, approximately distributed as the Normal Standard Distribution, see §A.4.3.2:

$$z_W \dot{\sim} \mathcal{N}(0, 1)$$

Step 5, finally, computes the p-value p_W associated to z_W (and then to W), according to Table 2.2 for the two-tailed model.

6.3.2 DECT-Q Test

The definition of the DECT-Q Test test is given by Table 6.4. Steps 1, 2 and 3 are

self-explanatory and end with the computation of the set of z_j values, according to Equation (6.7), and of Q as the maximum absolute value among all the z_j values.

Step 4, finally, computes the p-value p_Q associated to Q, according to the following observations. First, Q can be written as

$$Q = max(Q', Q'') \tag{6.16}$$

with

$$Q' = \max_{z_j<0}(-z_j)$$

$$Q'' = \max_{z_j\geq 0}(z_j)$$

Thus, Q' and Q'' represents the extreme left and right values of z_j, corresponding to the minimum value and the maximum value of K_j, respectively. Then, under the null hypothesis, the probability distributions of Q' and Q'' are identical and can be approximated by the Gumbel distribution, since they can be seen as the maximum of a set of independent and identically (normally) distributed variables (see §A.4.4):

$$Q', Q'' \dot{\sim} G(\mu_Q, \beta_Q) \tag{6.17}$$

with

$$\mu_Q = F^{-1}_{\mathcal{N}(0,1)}\left(1 - \frac{1}{2^\lambda}\right)$$

$$\beta_Q = F^{-1}_{\mathcal{N}(0,1)}\left(1 - \frac{1}{e2^\lambda}\right) - \mu_Q$$

where μ_Q and β_Q are the location and the scale of G, respectively.

According to Table A.4, the CDF of the Gumbel distribution in Equation (6.17) is

$$F_G(q) = e^{-e^{-\frac{(q-\mu_Q)}{\beta_Q}}} \tag{6.18}$$

In order to compute the p-value associated to Q, p_Q, that is, the probability that a value more extreme (higher) than Q is observed, we note that Q' and Q'' can be considered statistically independent, identically distributed random variables. As such, the CDF of their maximum is equal to the product of their CDFs (see [55], p.4, §1.3).

$$F_Q(q) = F_{Q'}(q)F_{Q''}(q) = F_G(q)^2$$

Thus, according to Table 2.2 for the right-tailed model,

$$p_Q = 1 - F_G(Q)^2$$

Defining, for the sake of easy reading, the auxiliary variable

$$\gamma_Q = e^{-e^{-\frac{(Q-\mu_Q)}{\beta_Q}}}$$

from Equation (6.18) we finally obtain

$$p_Q = 1 - \gamma_Q^2$$

6.3.3 Remarks

Some methodological considerations follow.

First, we observe that the DECT-Q Test relies on the z_j values, which are computed by the Normal Standard Distribution approximation, as by Equations (6.7) and (6.8), and step 2 of Table 6.4. On the contrary, the DECT-W Test is based on the K_j values, which (for $N \leq 2^{24}$) are precisely determined through exact computation, as shown in Equations (6.3) and (6.4). Despite being apparently inconsistent, we believe that the choice to adopt two different methods makes sense and has the following motivation. The Q value produced by the DECT-Q Test depends only on one value, that is the maximum (absolute) of the K_j values; hence, a small error in each K_j value estimation results at most in a small error in determining Q. On the contrary, the W value produced by the DECT-W Test is a sort of average index of the behaviour of all the K_j values, counting those passing and failing the K-Test; thus, even a small evaluation error on the used values may have considerable impact on the computation of the overall index W. Therefore, consistently with Observations 19 and 20, it makes sense to use the (simpler) Normal Standard Distribution approximation for the z_j computation and the (more expensive) exact computation for the K_j computation. For completeness, it has to be noted that for $N > 2^{24}$ the Normal Standard Distribution approximation is used also to compute W (due to the computational complexity of the exact computation), however the resulting estimation is very good, as discussed in §6.2.2, and thus the potential errors are negligible.

All the claims just reported have been confirmed by extensive simulations (see §6.5 and especially §6.5.3).

Second, we explain the constraints given on n and λ in Equation (6.2). Both come from the the rule of thumb given in Equation (A.1), to apply the Normal Distribution approximation to the Binomial Distribution. In particular, the approximation of Equation (6.4) given in Equation (6.6) implies $2^{n-1} \geq 10$ and then $n \geq 5$, while Equation (6.15) implies $2^{\lambda}\alpha \geq 10$, with $\alpha = 0.1$, and then $\lambda \geq 7$. Moreover, we comment on the choice of requiring L, λ and (consequently) N to be powers of 2: the base 2 looks the more natural when dealing with bit sequences, but other tests can be defined with different bases (this will be considered again in §6.7).

Third, we elaborate a bit on the test model chosen for the K-Test, the DECT-W Test and the DECT-Q Test. Though the choice is in principle arbitrary, the rationale behind is that for the first two we look for a balanced behaviour of the analysed sequences, which is consistent with the two-tailed model for K_j and W, while for the third, which considers the extreme (maximum and minimum) values, the right-tailed model is best suited for Q.

6.4 Implementation

We have implemented (in C language) the DECT-W Test and the DECT-Q Test described in the previous sections. According to Equation (6.14), given an L-bit input sequence, with $L = 2^l$, the implementation runs both tests with the following values for λ:

$$\lambda = 7, 8, \ldots, l-5$$

Thus, the implementation is, in fact, a collection of H pairs of DECT-W Tests and DECT-Q Tests, with

$$H = l - 11 \tag{6.19}$$

As anticipated, the collection of these $2H$ tests is referred to as DECT Suite. The default output of DECT Suite provides the basic information, as for example in Figure 6.2, where the input file (*rand*[5]) has size $L = 2^{27}$ bits and then $\lambda = 7, 8, \ldots, 22$. For each value of λ (first column, λ), the corresponding p-values (truncated to 4

[5]Here it is not relevant how the file has been generated, since we are just describing the implementation interface.

```
File 'rand' of size 2^27 bits

 λ   |======== W =========|======== Q =========|
     |                    |                    |
 7   |       0.3458       |       0.8038       |
 8   |       0.5880       |       0.4068       |
 9   |       0.4795       |       0.2118       |
10   |       0.0838 W1    |       0.8576       |
11   |       0.2956       |       0.4401       |
12   |       0.5527       |       0.7231       |
13   |       0.0953 W1    |       0.6963       |
14   |       0.6084       |       0.4782       |
15   |       0.6744       |       0.5041       |
16   |       0.2587       |       0.4135       |
17   |       0.4810       |       0.8798       |
18   |       0.6234       |       0.9267       |
19   |       0.7474       |       0.9735       |
20   |       0.4238       |       0.0520 Q1    |
21   |       0.2565       |       0.9874       |
22   |       0.7716       |       0.9572       |
```

Fig. 6.2 Test output (I)

decimal places) are reported: p_W (second column, W) and p_Q (third column, Q), computed as in Tables 6.3 and 6.4, respectively. We observe that a $W1$ tag appears in the W column for $\lambda = 10$ and $\lambda = 13$, and that a $Q1$ tag appears in the Q column for $\lambda = 20$. These tags highlight that the corresponding p-values are less than 10^{-1}. More in general, each time a p-value (either p_W or p_Q) falls in the range $[10^{-(t+1)}, 10^{-t})$ for an integer t, $1 \leq t \leq 8$, an analogous tag is reported with index t. If the p-value falls in the range $(-\infty, 10^{-9})$, then tags $W9$ and $Q9$ are used (we believe that a more fine-grained resolution would be useless: there is no practical difference in knowing that the *a priori* probability, according to the null hypothesis, of an observed value is equal to 10^{-9} or, instead, even smaller: the null hypothesis has certainly to be rejected in either case). For example, in Figure 6.3, tags $W2, W3, Q2, Q3, Q4, Q7$ and $Q9$ appear as well (since only 4 decimal places are reported for each p-value, $Q4, Q7$ and $Q9$, and in general each tag ≥ 4, are indistinguishable).

The goal of the tags is simply to provide a quick visual evidence that the observed p-values may indicate a discrepancy with the assumed null hypothesis. In this regard, some comments are however presented below.

```
File 'seq48_0.bin' of size 2^26 bits

 λ   |======== W =========|======== Q =========|
     |                    |                    |
 7   |       0.0875 W1    |       0.9610       |
 8   |       0.4533       |       0.5256       |
 9   |       0.8597       |       0.5012       |
10   |       0.6467       |       0.5694       |
11   |       0.8945       |       0.7275       |
12   |       0.6171       |       0.0000 Q7    |
13   |       0.2820       |       0.0001 Q3    |
14   |       0.8044       |       0.0000 Q4    |
15   |       0.6298       |       0.0698 Q1    |
16   |       0.1365       |       0.0000 Q9    |
17   |       0.0003 W3    |       0.0000 Q4    |
18   |       0.0612 W1    |       0.0025 Q2    |
19   |       0.0035 W2    |       0.1534       |
20   |       0.7202       |       0.8819       |
21   |       0.6337       |       0.7884       |
```

Fig. 6.3 Test output (II)

First, we observe that, as already discussed in §2.1.3 and §4.4.1.1, if the null hypothesis is correct, then occasional failures (that is, low p-values and then high tag indexes) are not only acceptable, but expected. More precisely the probability that a given sequence for a given λ determines a tag with index t is approximately[6] f_t, with

$$f_t = 10^{-t} - 10^{-(t+1)} \tag{6.20}$$

Thus, for each column (W, Q), it is not surprising to observe one or a few tags with index 1 (that is, $W1$ or $Q1$): such a situation does not contradict the null hypothesis, since each column contains 16 elements and $f_t \approx 0.1$. On the contrary, finding tags with high indexes, like in Figure 6.3, strongly supports the rejection of the null hypothesis: for example for $\lambda = 12$ we find the $Q7$ tag, meaning that the observed p-value is expected to appear on average just about once over 10 millions times *if the null hypothesis is satisfied.*

[6]The actual value can be different since we are in the discrete setting, which has been analyzed in Chapter 3.

Second, if all the values in each column were independent, then we would precisely compute the number E_t of expected tags with index t as

$$E_t = f_t H = (10^{-t} - 10^{-(t+1)})(l - 11)$$

(see Equations (6.19) and (6.20)). We observe however that all the values in each column derive from the same sequence and, hence, are dependent on each other. In particular, looking at the definition of the tests it is clear that the closer the values of λ, the stronger the relation between the corresponding p-values (and, then, of the tags)[7]. It is therefore not surprising that statistical anomalies (or, equivalently, tags with high indexes) tend to appear in clusters (like $\lambda \in [17, 19]$ for the W column and $\lambda \in [12, 18]$ for the Q column in Figure 6.3).

Finally, we note that outputs reported in Figures 6.2 and 6.3 are the basic report form of the tests implementation. Options for more detailed outputs, including all the intermediate values as defined in Tables 6.3 and 6.4, are also available, but here we skip their description since we are more focused on the conceptual level.

6.5 Validation

Both the definition and the implementation of a hypothesis test can conceal many pitfalls, as well discussed in [17], §4.3. Issues can be very subtle and hard to recognize, so a strong validation strategy is necessary. In this section we propose a validation methodology and describe its application to the DECT-W Test and the DECT-Q Test. In particular, in §6.5.1 we outline a general validation procedure, which we apply first in §6.5.2, following a (questionable) NIST recommendation, and then in §6.5.3, based on a different rationale.

6.5.1 Methodology

In its typical use, a hypothesis test, T, allows to draw a (probabilistic) conclusion about the correctness of a certain null hypothesis, H_0, on the data distribution of the random generation process, based on the analysis of a set of input data: if data are

[7] Investigation of the exact relation between the p-values obtained for different λ is left as future work.

consistent with H_0, then H_0 is considered true, otherwise it is deemed false. Doing this, we obviously make the *assumption* that the test itself, T, is correct[8].

Hereinafter we refer to the standard process just described as *direct validation procedure*. An example of such a procedure is given by NIST ([17] at p. 4-1, §4.1) and is summarized in Table 5.2, whilea high-level description of the direct procedure is given in Table 6.5.

Given a hypothesis test T and a null hypothesis H_0, in order to validate H_0

1. Assume T is correct;
2. observe a large number of samples (for example, sequences);
3. apply the hypothesis test T to the observed samples of step 2;
4. verify that the results obtained in step 3 are consistent with those expected based on the theoretical analysis of the hypothesis test, T, and of the null hypothesis, H_0;
5. if step 4 is successful, then conclude that the null hypothesis, H_0, is true. Otherwise, that it is false.

Table 6.5 Direct validation procedure

Conversely, if the goal is to validate the test (both the definition and the implementation), we can invert the point of view: rather than employing a (correct) test, T, to validate a null hypothesis, H_0, about the data, we can use input data generated according to H_0 to validate the correctness of the test T. The role of the *assumption* and of the *null hypothesis* are, thus, switched compared to the direct validation procedure: the resulting process is still a hypothesis test, but the null hypothesis, V_0, is that the test T is correct, while we *assume* that input data are generated according to (the data distribution underlying) H_0.

[8] Henceforth, sometimes for simplicity we say "correct" instead of "correctly defined and implemented", keeping however in mind that implementation issues are as important and error-prone as theoretical ones.

Hereinafter we refer to the process just described as *inverse validation procedure*[9].

Moving to our specific setting of binary sequences randomness tests, the null hypothesis in the inverse validation procedure is that a given test is correct and the assumption is that all the input sequences are random[10]. Thus, applying the inverse validation procedure means that, instead of using a (correct) test to validate the randomness of the sequences, we use random sequences to validate the correctness of the test.

Thus, a high-level description of the inverse procedure is given in Table 6.6.

Given a hypothesis test T with an associated null hypothesis H_0, in order to validate the null hypothesis V_0 that the test is correct

1. Assume input data follow the data distribution underlying H_0, that is all the input sequences are random;
2. take a large number of random sequences;
3. apply the hypothesis test T to the random sequences of step 2;
4. verify that the results obtained in step 3 are consistent with those expected based on the theoretical analysis of the hypothesis test, T;
5. if step 4 is successful, then conclude that the hypothesis test, T, is correct. Otherwise, that it is not correct.

Table 6.6 Inverse validation procedure

We emphasize that the inverse validation procedure described in Table 6.6 is basically the same as the direct one described in Table 6.5, with two slight (but fundamental) differences, shown in Table 6.7.

[9]The distinction between *direct* and *inverse* validation procedure is not standard, but here it turns out to be a useful terminology.

[10]In fact, as discussed in §5.2.1, we should more properly speak of sequences randomly generated according to the uniform distribution. Here, however, for the sake of clarity we conform to the common simplification of referring to random sequences.

- in step 2, the input sequences in the inverse procedure are taken accordingly to the theoretical null hypothesis, H_0 (that is, in our case, sequences are random), while in the direct procedure they are the actual observed data;
- in step 5, in the inverse procedure the conclusion is drawn on the correctness of the hypothesis test, while in the direct procedure it is drawn on the correctness of the null hypothesis, H_0.

Table 6.7 From direct to inverse validation procedure

If the null hypothesis of the inverse validation procedure, V_0, is satisfied (that is, the test is correct), then the procedure succeeds, up to occasional failures (Type I Errors). Conversely, if V_0 is not satisfied, then the validation procedure is likely to fail[11], even if it can occasionally be successful as well (Type II Errors). As with any hypothesis test, in order to be useful, both the Type I Error probability and the Type II Error probability of the inverse validation procedure should be as low as possible. Estimation of Type I Error probability for the inverse procedure is in principle straightforward, as it coincides with that of the direct procedure, since the two settings are coincident by construction[12]. However, the Type I Error probability of the direct procedure is often not easy to determine, as seen in §5.2.2.2, where we show that the value proposed by NIST in Table 5.3 ($\alpha'' = 0.0001$) is questionable. Moreover, Type II Error probability is hard to compute and, in practical cases, it is in fact impossible, as discussed in §2.1.4. Though, the inverse validation procedure can

[11] We observe that, when this happens, it does not provide a clear indication of where the error lies, but it tells that somewhere in the procedure there likely is an hidden issue (either logical or implementation-related, possibly even in the definition or implementation of the validation procedure itself) which should be investigated.

[12] When it comes to evaluating the Type I Error, both the direct and the inverse validation procedures assume that two conditions are satisfied: the test is correct (that is V_0) and the input sequences are random. On the contrary, when evaluating the Type II Error, the direct procedure assumes that the condition on the correctness of the test is valid, while the condition on the randomness of the input sequences represents the null hypothesis and is, therefore, assumed false. On the other hand, the inverse procedure assumes that the condition on the randomness of the input sequences is valid, while the condition on the correctness of the test represents the null hypothesis and is, therefore, assumed false.

still be useful, but requires a more qualitative approach (as we see later, in §6.5.2 and §6.5.3).

Relying on the above considerations, and in particular to Tables 6.6 and 6.7, we can make the following

Observation 21. *Every direct procedure (designed to validate a given hypothesis test) can be converted into an inverse procedure (aimed at validating the test itself), applying the (small) changes reported in Table 6.7.*

The test validation procedure described in Table 6.6 looks then very simple. However, there is a fundamental conceptual problem here: we need random sequences to validate the correctness of the tests necessary to validate the randomness of the sequences. Apparently, it's a dog chasing its own tail. Furthermore, to complicate matters, the concept of randomness itself is very elusive by nature and is therefore not easy to define. Indeed, as we have seen in the previous chapters, statistical tests can provide strong evidence that a generation process is not (sufficiently) random, but, on the contrary, they can never guarantee that a given process is perfectly random.

So, what we can concretely do is to to take sequences for which we have high confidence that they behave (in any reasonable concrete sense) as if they were randomly generated[13]. For this aim, we use the Advanced Encryption Standard (AES)-based generation mechanism described in §A.7, that is, AES-GCM. Feeding AES-GCM with distinct (key, nonce) pairs, we obtain a (virtually) unlimited number of sequences which appear to be generated by a uniform random process[14]. We emphasize the conceptual step: we just have to guarantee that the input pairs are distinct (which is easy to do, no randomness is required) and the AES-GCM mechanism provides a set of corresponding sequences that can be considered (practically) random and independent. The reliability of this approach lies in the confidence that the entire cryptographic community places on the indistinguishability of the sequences (keystreams) produced by AES-GCM compared to those produced by an ideal random uniform process.

[13]More technically, we assume there is no (concrete) way to build a distinguisher, that is, an algorithm able to detect in the sequence a statistical anomaly with respect to ideal random data.

[14]Incidentally, we observe that using pseudo-random sequences instead of truly random ones has an advantageous side effect, that is, in the presence of unclear experimental results, it is always possible to precisely replicate the experiment to gain a better understanding of the observed data: we just have to store the (short) (key, nonce) pairs determining the sequences, instead of the (impracticably long) sequences themselves.

6.5.2 A first (unsuccessful) attempt

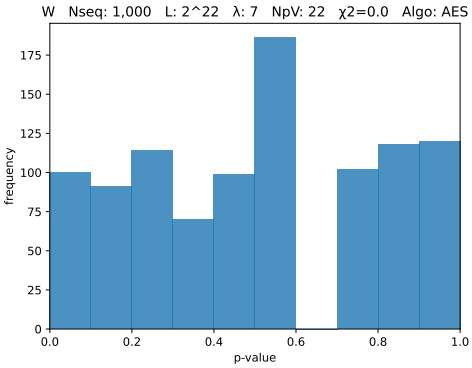
Once completed the DECT-W Test and DECT-Q Test definitions and implementations, our chronologically first validation attempt was to build an inverse validation procedure (to validate the two tests), applying Observation 21 to the *NIST procedure* proposed in [17] (p. 4-3, §4.2.2), which we have reported in Table 5.3 and commented in §5.2.2.2. Given a hypothesis test, the NIST procedure consists in taking a set of sequences, deriving the corresponding p-values and verifying that they are uniformly distributed. With this aim, two methods are suggested: first, to build a histogram of the frequencies of the p-values and visually check their distribution; second, to apply a χ^2 test to the collected p-values.

Relying on the procedure just recalled, we implemented the inverse validation procedure described in Table 6.6. Results are shown in Figure 6.4, obtained from a set of $1,000$ sequences (hereinafter, unless otherwise stated, random sequences are generated by AES-GCM, as anticipated in §6.5.1), each $L = 2^l$-bit long, with $l = 22$, thus determining the λ range as $[7,17]$, according to Equation (6.14). The figure contains the histograms of the p-values obtained for both the DECT-W Test and the DECT-Q Test, for $\lambda = 7,12,17$ (that is, the two extreme and an intermediate values, as significant example settings). In addition, in the header of each figure, besides the obvious parameters, we include the number NpV of different p-values found among the $1,000$ considered sequences (then $1 \leq NpV \leq 1000$) and the corresponding χ^2 p-value, computed according to the NIST procedure[15].

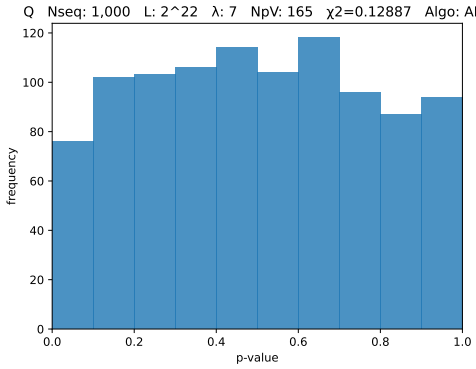
Table 6.8 extends the observed values to any $\lambda \in [7,17]$. Unfortunately, results are discouraging. Even with a simple visual qualitative inspection, it is immediately clear that most distributions are far from uniformity (Figures 6.4a and 6.4f are particularly evident cases, but also Figures 6.4c and 6.4d clearly show a bad profile).

A quantitative analysis of Table 6.8 is even more convincing. We note that most χ^2 p-values are very close to 0 and that, even generously setting the acceptance threshold to $\alpha'' = 0.0001$, as suggested in [17] (p. 4-3, §4.2.2), the NIST procedure is successful only for a minority of the λ values, namely $\lambda = 10$ and $\lambda \geq 13$ for the DECT-W Test and $\lambda \leq 9$ for the DECT-Q Test, while all the other values, highlighted in red, are less than α'' (notice that, in Table 6.8, every value in the range $(0, 10^{-5})$ is indicated as $< 10^{-5}$ because the precision of the values reported in the table is 5

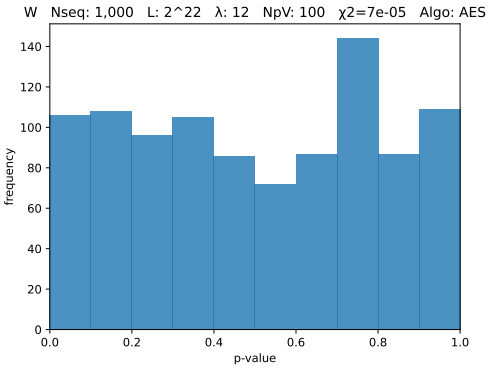
[15]The χ^2 *p-value* is abbreviated in the figures as χ^2 for space reasons. Also notice that any χ^2 p-value less than 10^{-5} appears as 0.0.



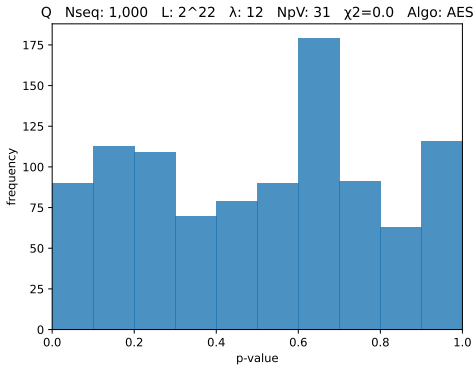
(a) $\lambda = 7$, DECT-W Test



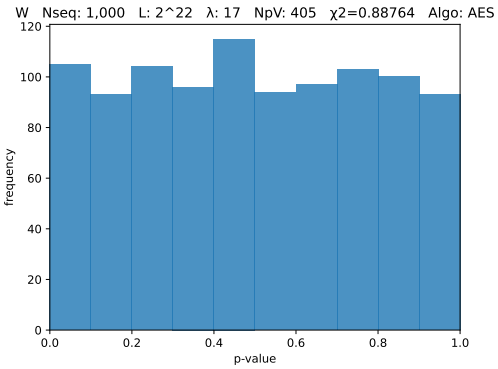
(b) $\lambda = 7$, DECT-Q Test



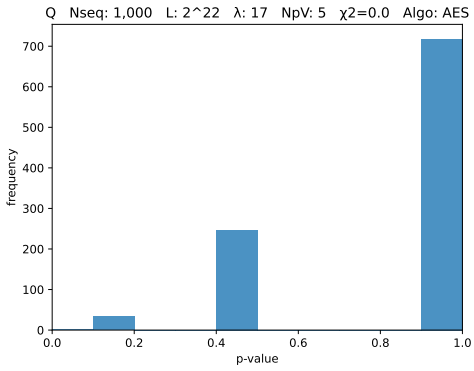
(c) $\lambda = 12$, DECT-W Test



(d) $\lambda = 12$, DECT-Q Test



(e) $\lambda = 17$, DECT-W Test



(f) $\lambda = 17$, DECT-Q Test

Fig. 6.4 P-value histograms, $\lambda = 7, 12, 17$

	DECT-W Test		DECT-Q Test	
λ	NpV	$\chi^2 pv$	NpV	$\chi^2 pv$
7	22	$< 10^{-5}$	165	0.12887
8	30	$< 10^{-5}$	116	0.28123
9	44	$< 10^{-5}$	83	0.00125
10	59	0.00098	61	$< 10^{-5}$
11	76	$< 10^{-5}$	42	$< 10^{-5}$
12	100	0.00007	31	$< 10^{-5}$
13	143	0.67247	22	$< 10^{-5}$
14	180	0.35864	15	$< 10^{-5}$
15	233	0.37701	12	$< 10^{-5}$
16	300	0.12520	6	$< 10^{-5}$
17	405	0.88764	5	$< 10^{-5}$

Table 6.8 Number of distinct p-values and χ^2 p-value, $\lambda \in [7, 17]$

decimal places). The failures reported motivated us to conduct a thorough analysis of the discrete setting, which led to the writing of Chapter 3. Based on the results obtained in the mentioned chapter, we realized that the issue did not lie in the tests definition and implementation, but rather in the logic of the NIST procedure, as below examined.

Indeed, from Table 6.8 we note that the number of distinct observed p-values (NpV) varies monotonically with λ (increasing for the DECT-W Test and decreasing for the DECT-Q Test). In both cases, when NpV is small, the corresponding χ^2 p-values are generally close to 0 and the NIST procedure fails. In light of the results obtained in §3.4.4, this is not surprising, since the requirements for the application of a χ^2 test (and, more in general, of a Goodness-of-Fit test), based on the alleged uniformity of the p-values, are not met. In particular, Table 3.3 requires K to be very small with respect to N_Ω, where K is the number of sub-intervals in the χ^2 test and N_Ω is the cardinality of the set (Ω) of p-values associated to the test under validation (namely, the DECT-Q Test and the DECT-W Test). In our case (that is, the χ^2 test of the NIST procedure, see Table 5.3), however, the condition is not met for most values of λ, as shown in Table 6.8, where we have $K = 10$ and $N_\Omega \approx NpV$ [16].

[16] Rigorously, we have $N_\Omega \geq NpV$, since N_Ω is the number of possible p-values, while NpV is the number of observed p-values. However, since the number of observed sequences (1,000) is significantly higher than NpV, we do not reasonably expect that the difference between N_Ω and NpV is significant.

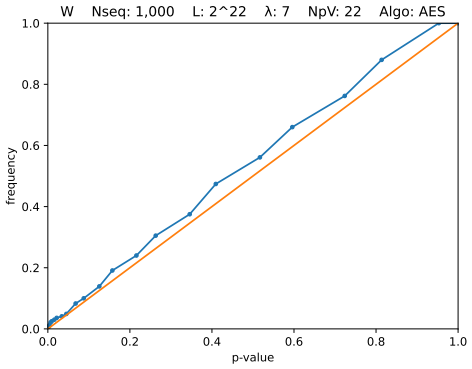
6.5.3 A second (more satisfactory) attempt

Once realized that the validation procedure of Table 5.3 is not always reliable, we took another approach, based on Equation (3.3) which here we recall

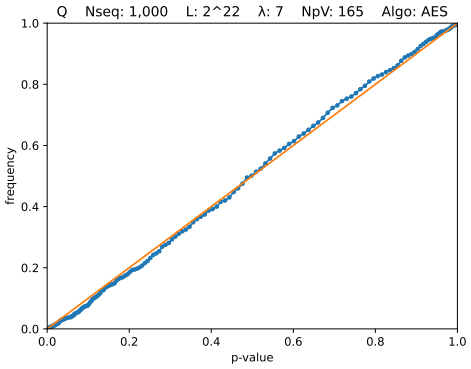
$$Pr(PV \leq \omega | H_0) = \omega, \forall \omega \in \Omega$$

As observed in §3.2, the equality holds only for $\omega \in \Omega$, that is, only for the p-values determined by the definition of the test under validation.

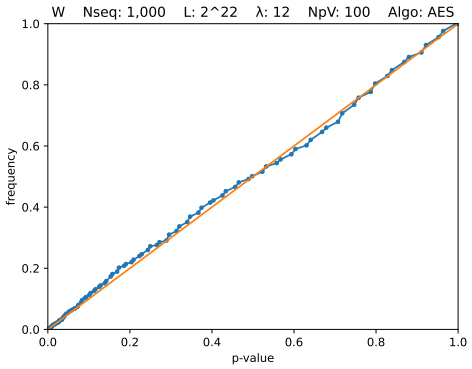
Equation (3.3) implies that all the N_Ω points of the set $\Omega^* = \{(\omega, F_{PV}(\omega)), \omega \in \Omega\}$, with F_{PV} being the CDF of the p-value, lay on the identity line $y = x$. Hence, considering the same setting of §6.5.2 (that is $1,000$ random and independent sequences, each 2^{22}-bit long) we have plotted Ω^* for each $\lambda \in [7,17]$. Visual analysis of the obtained graphs confirms the expected behaviour, with the p-value CDF points substantially overlapping the identity line $y = x$ (with stronger visual evidence where the number of observed p-values, NpV, is higher). For example purposes, in Figure 6.5 we report (in blue) the graphs obtained for $\lambda = 7, 12, 17$ for both the DECT-W Test and the DECT-Q Test (whereas the identity line is in orange).



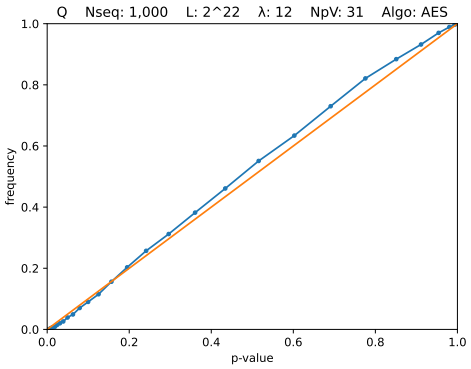
(a) $\lambda = 7$, DECT-W Test



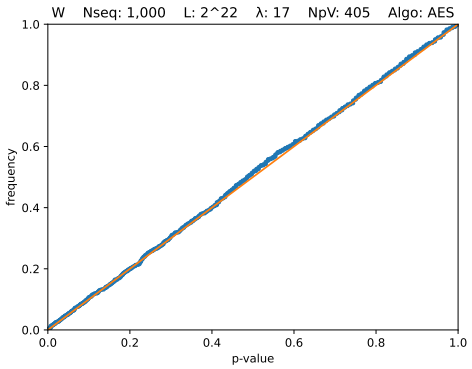
(b) $\lambda = 7$, DECT-Q Test



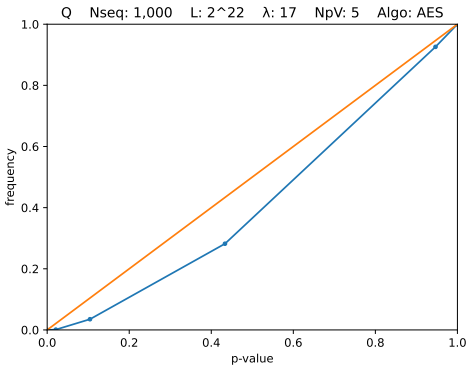
(c) $\lambda = 12$, DECT-W Test



(d) $\lambda = 12$, DECT-Q Test



(e) $\lambda = 17$, DECT-W Test



(f) $\lambda = 17$, DECT-Q Test

Fig. 6.5 P-value CDF, $\lambda = 7, 12, 17$

As already discussed at a higher level in §6.5.1, the visual test above described can be seen as a (qualitative) hypothesis test[17], where the null hypothesis, V_0, is given in Table 6.9.

the DECT-W Test and the DECT-Q Test are correctly defined and implemented.

Table 6.9 Null Hypothesis, V_0, DECT Suite

Extensive simulations (exemplified here by Figure 6.5) give us good evidence that, when the null hypothesis V_0 (Table 6.9) is satisfied, our validation procedure, applied to the DECT-W Test and the DECT-Q Test, is successful, that is, the p-value CDF is well approximated by the identity line (in $[0,1]$).

As discussed in §6.5.1, in order to gain more confidence in the correctness of our two tests, we are also interested in checking that, when the null hypothesis (Table 6.9), V_0, is not satisfied, the validation procedure is, consistently, likely to fail. Hence, here we invalidate V_0 (Table 6.9), providing an incorrect test p-value computation, in §6.5.3.1 for the DECT-W Test and in §6.5.3.2 for the DECT-Q Test. Moreover, in §6.5.3.3, we analyse the setting where the input sequences are non-perfectly-random (thus violating the assumption made in Table 6.6).

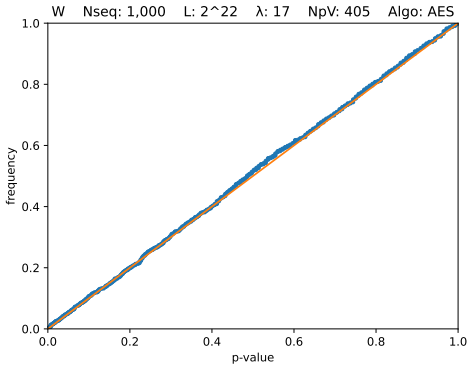
6.5.3.1 Incorrect DECT-W Test definition

The DECT-W Test is defined in Table 6.3. In particular, in step 3, the parameter α is approximated to 0.1 for $n > 24$, in view of the good approximation provided by the Normal Distribution to the Binomial Distribution. Conversely, for $n \leq 24$ the exact computation is preferred, because the above-mentioned approximation is not good enough for small values of n, as discussed throughout §6.2.

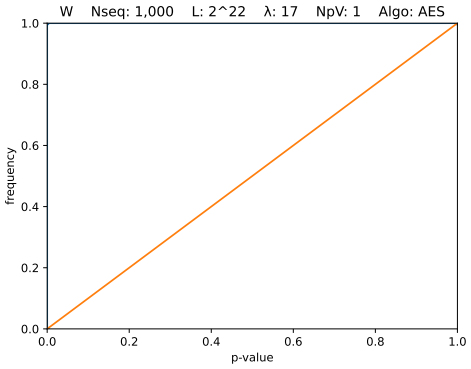
In order to (slightly) alter the p-value computation, we have changed the definition of the DECT-W Test, setting $\alpha = 0.1$ for any n. In the same setting and with the same 1,000 sequences as in §6.5.3, we have run experiments for $\lambda \in [7,17]$ for the

[17] Strictly speaking, a hypothesis test is by nature quantitative. The difficulty in computing a precise test statistic here is that, in general, we do not know the full p-values set Ω, but only the subset of p-values we have actually observed. So we limit to a more qualitative, but still meaningful, visual check.

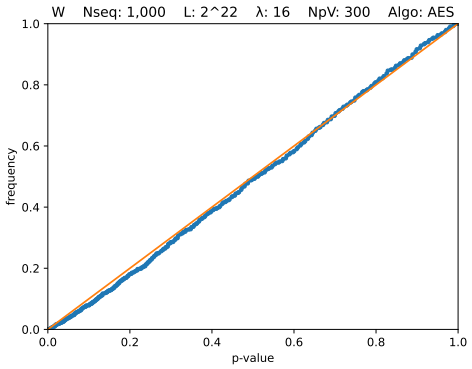
DECT-W Test. As expected, the biggest discrepancies with the results obtained with the exact version of the test (reported in §6.5.3) occur for those values of λ for which the corresponding α is farthest from 0.1, see Table 6.1. We observe that the worst cases are, not surprisingly, for the smallest values of n, while the best results, with α values only negligibly far from 0.1, can be found for the highest values of n. Recalling that $\lambda = l - n$ according to Equation (6.1), in Figure 6.6 we report some example graphs of the resulting p-value CDF for different values of λ (on the right side), together with the corresponding graphs obtained from the correct test implementation (on the left side), maintaining the same layout of Figure 6.5.



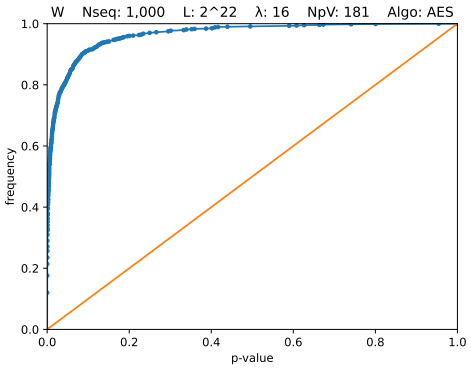
(a) $\lambda = 17$, correct definition



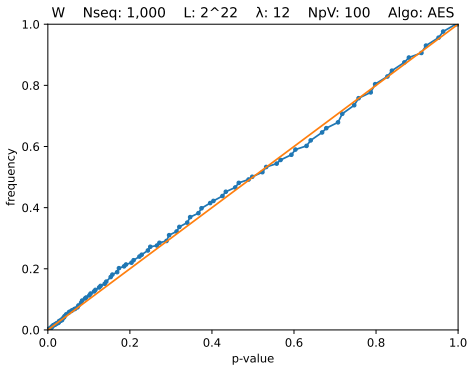
(b) $\lambda = 17$, altered definition



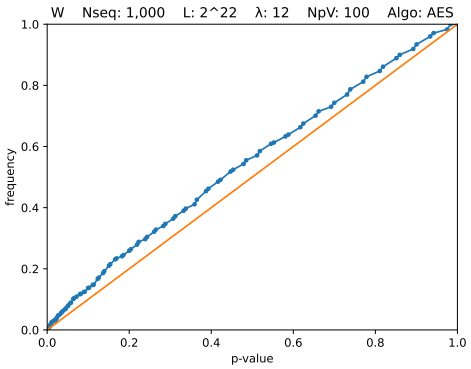
(c) $\lambda = 16$, correct definition



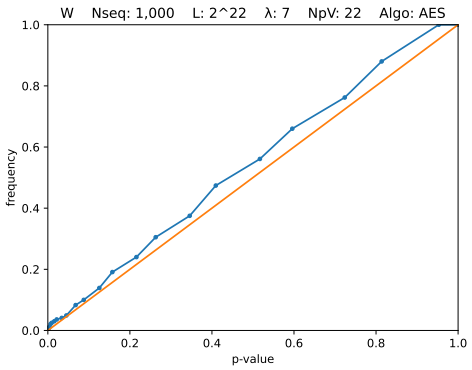
(d) $\lambda = 16$, altered definition



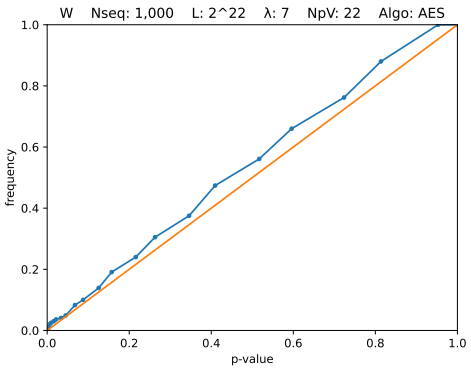
(e) $\lambda = 12$, correct definition



(f) $\lambda = 12$, altered definition



(g) $\lambda = 7$, correct definition



(h) $\lambda = 7$, altered definition

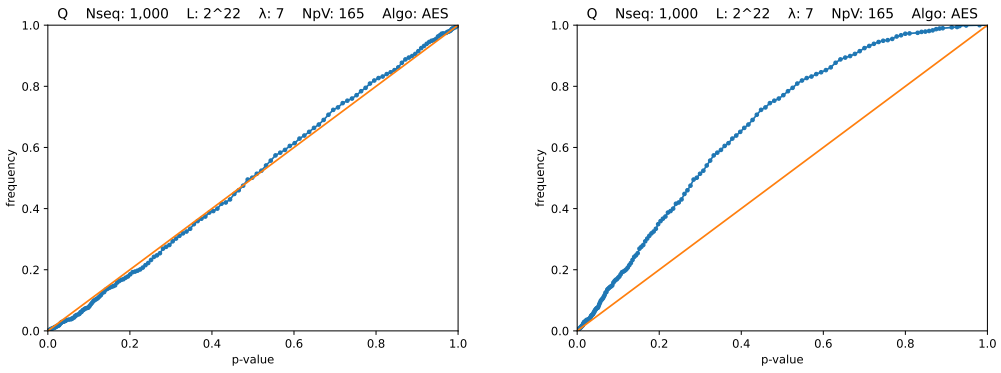
Fig. 6.6 DECT-W Test p-value CDF, $\lambda = 17, 16, 12, 7$, correct and altered definitions

As expected, we see that for high values of λ ($\lambda = 17$, Figures 6.6a, 6.6b, and $\lambda = 16$, Figures 6.6c, 6.6d), the two curves are significantly different (in Figure 6.6b the function even collapses in a single dot in the upper left corner of the graph), whereas for the intermediate value of λ ($\lambda = 12$, Figures 6.6e, 6.6f) the difference is still present but smaller and finally for the lowest value of λ ($\lambda = 7$, Figures 6.6g, 6.6h) the two curves coincide. Overall, observing the high values of λ (that is, the low values of n) we can conclude that, given the approximated version of α values in Table 6.3, the validation procedure fails, confirming the importance of using, instead, the exact computation of Table 6.1 for low values of n.

6.5.3.2 Incorrect DECT-Q Test definition

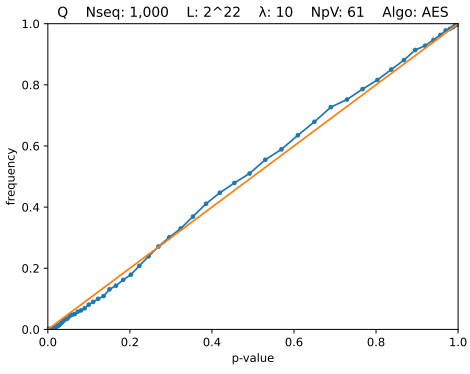
Similarly, we show that an incorrect computation of the p-value in the DECT-Q Test, defined in Table 6.3, leads to evident disagreement in the p-value CDF if compared with the correct definition. To alter the definition of the p-value computation, we (arbitrarily) replace $p_Q = 1 - \gamma_Q^2$ with $p_Q = 1 - \gamma_Q$ in step 4 of Table 6.4[18]. Extensive simulations have been done for different values of λ and results are reported in Figure 6.7 for $\lambda = 7, 10, 13, 17$ (that is, the extreme and two intermediate values), using again the same layout of Figures 6.5 and 6.6.

[18]This modification corresponds to substitute $Q = \max(|z_j|)$ with $Q = \max(z_j)$ in step 3 of the same table. In particular, in Equation (6.16) we are replacing $Q = max(Q', Q'')$ with $Q = Q''$.

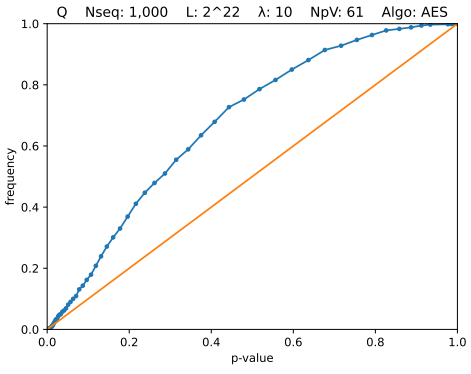


(a) $\lambda = 7$, correct definition

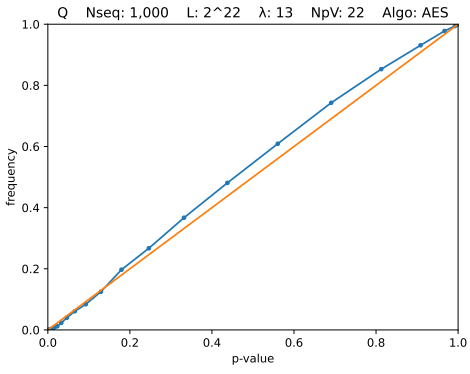
(b) $\lambda = 7$, altered definition



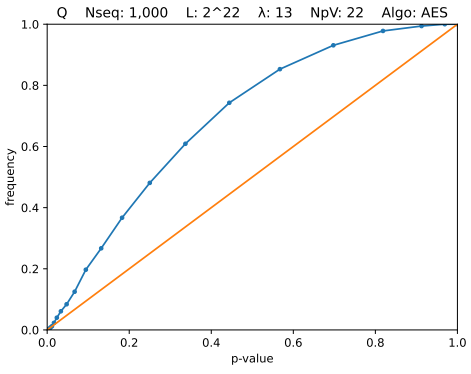
(c) $\lambda = 10$, correct definition



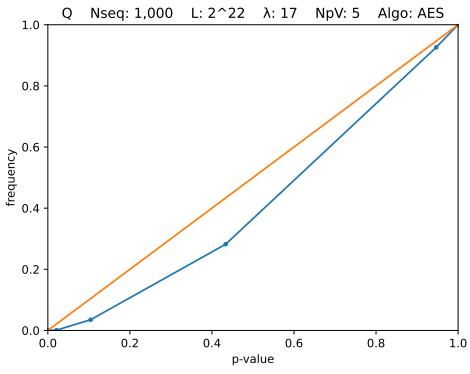
(d) $\lambda = 10$, altered definition



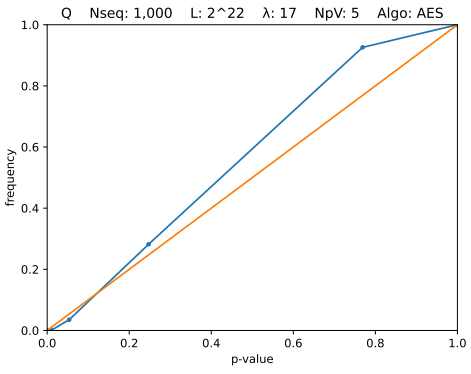
(e) $\lambda = 13$, correct definition



(f) $\lambda = 13$, altered definition



(g) $\lambda = 17$, correct definition



(h) $\lambda = 17$, altered definition

Fig. 6.7 DECT-Q Test p-value CDF, $\lambda = 17, 16, 12, 7$, correct and altered definitions

Analysis has revealed that for any λ the discrepancy with the correct version of the DECT-Q Test is evident, as shown for $\lambda = 7$ (Figures 6.7a and 6.7b), $\lambda = 10$ (Figures 6.7c and 6.7d) and $\lambda = 13$ (Figures 6.7e and 6.7f). An exception is observed for $\lambda = 17$ (Figures 6.7g and 6.7h), probably due to the small number of p-values (which tends to decrease as λ increases, as already observed)[19].

6.5.3.3 Non-random sequences

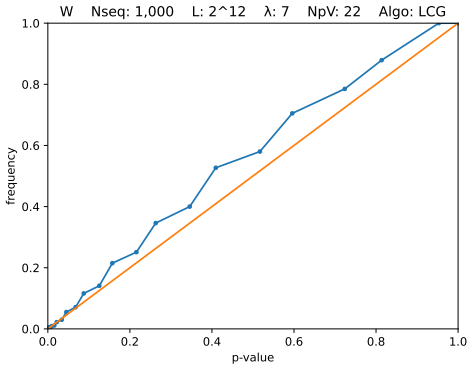
In §6.5.3.1 and §6.5.3.2 we have seen that our validation procedure is able to detect, at least to some extent, imprecise test definition (or implementation), that is a violation of the null hypothesis given in Table 6.9. Here, we also want to verify the relevance of the assumption made in Table 6.6, namely that the input sequences are random. Thus, we feed the validation procedure with non-random sequences, using however the correct versions of the DECT-W Test and the DECT-Q Test. In the same setting as before, we still consider $1,000$ sequences, each 2^{22}-bit long, but we produce them with a PRNG which is considered statistically good but not ideal (and, hence, not suitable for cryptographic use), that is the Microsoft Visual C++ LCG[20], initialized with $1,000$ different seeds. Also in this case, results of the simulations unambiguously indicate a failure of the validation procedure. Figure 6.8 reports the output of the two tests for $\lambda \in [7,10]$. Although with a couple of borderline cases (Figures 6.8a and 6.8e), on the whole, graphs of Figure 6.8 give strong evidence that the validation procedure fails in this case, where the input sequences do not satisfy the randomness assumption. For $\lambda > 10$ graphs of both tests even collapse in a single dot in the upper left corner and, thus, are not reported here.

6.5.3.4 Conclusions

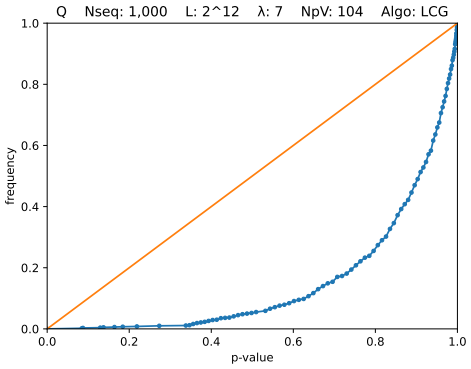
Summarizing, in this section (§6.5.3) we have considered a concrete realization of the high-level validation procedure presented in §6.5.1. First we have verified that, when the underlying null hypothesis on the test itself (Table 6.9) is satisfied,

[19] A future work should analyse how much the Gumbel approximation in Equation (6.17) is correct for high values of λ.

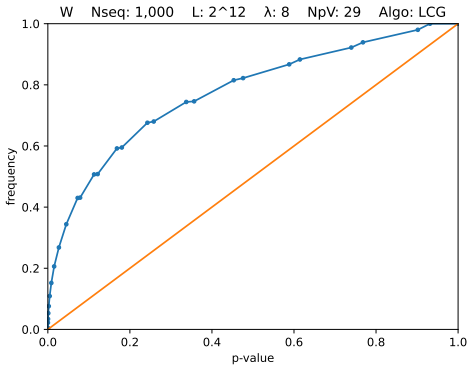
[20] A more detailed description of the LCGs is given in §A.8. Here it suffices to know that an LCG is a PRNG initialized by a seed and is defined by a set of three parameters: a multiplicative constant, an additive constant and a modulus, which in the specific case of the Microsoft LCG are $A = 214013, B = 2531011$ and $m = 2^{31}$, respectively.



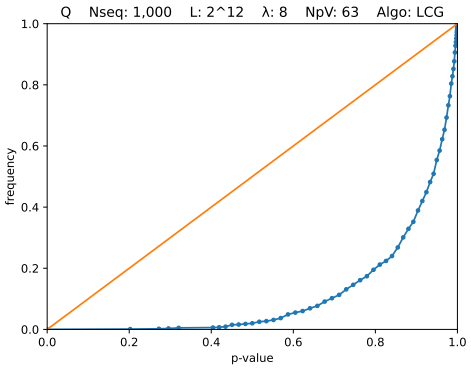
(a) $\lambda = 7$, DECT-W Test



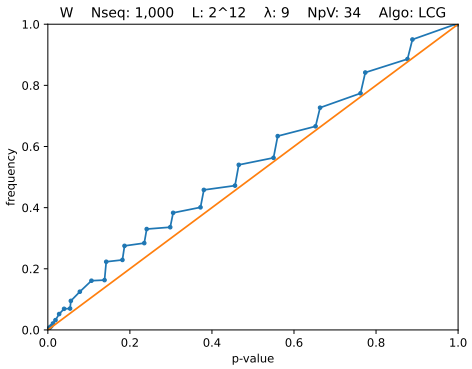
(b) $\lambda = 7$, DECT-Q Test



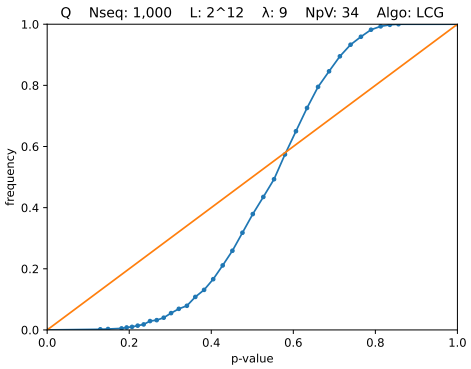
(c) $\lambda = 8$, DECT-W Test



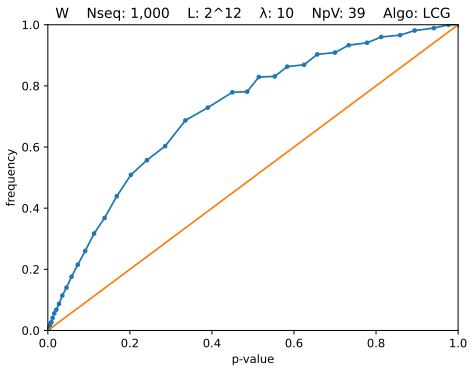
(d) $\lambda = 8$, DECT-Q Test



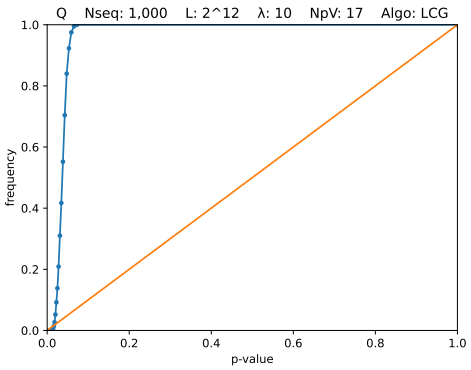
(e) $\lambda = 9$, DECT-W Test



(f) $\lambda = 9$, DECT-Q Test



(g) $\lambda = 10$, DECT-W Test



(h) $\lambda = 10$, DECT-Q Test

Fig. 6.8 LCG sequences, p-value CDF, $\lambda \in [7, 10]$

then the validation procedure of the test is successful. Then, we have also shown that (slightly) invalidating the null hypothesis or other underlying assumption (the randomness of the sequences) can be enough to make the validation procedure fail. In particular we have considered three different kinds of invalidation:

- in §6.5.3.1 we have invalidated the DECT-W Test definition, replacing the exact computation of the acceptance region with its standard approximation.
- in §6.5.3.2 we have invalidated the DECT-Q Test definition, replacing the computation of the p-value of the maximum absolute value of all the values with that of the maximum value among the positive values.
- in §6.5.3.3 we have invalidated the random sequences generation process, replacing the set of AES-based random sequences with a set of (non ideally random) LCG-based ones.

We emphasize that all the mentioned modifications appear as minor ones (or even legitimate). Nevertheless, we have shown that in all three cases the validation procedure fails, thus proving to be quite sensitive to deviations from the null hypothesis and other associated assumptions. Of course, it is possible that other discrepancies from the null hypothesis can go undetected, but on the basis of our experiments we feel quite confident concluding that the DECT-W Test and the DECT-Q Test are correctly defined and implemented (and, thus, also the DECT Suite, which collects $H = l - 11$ instances for each test, see Equation (6.19)).

6.6 Linear Congruential Generators analysis

After its validation, we have applied the DECT Suite to many commonly used TRNGs and PRNGs, comparing the results with those obtained with the NIST-STS suite proposed by NIST (see §5.2) and with other suites (collections) of tests (see §5.1). As expected for any suite (or single test), the DECT Suite is more effective on some classes of generators and less on others. In particular, in an ongoing joint work with Edoardo Signorini, we have used the DECT Suite on many different LCGs (see §A.8) and observed that it performs very well with most pa-

rameter choices, including those reported in Table A.5[21]. In §6.6.1 we describe the application of the DECT Suite on the Microsoft Visual C++ LCG and compare the results with those obtained by applying the NIST-STS on the same generator. Then, in §6.6.2, we extend the analysis to generators with much larger parameters and consider other test suites as well.

6.6.1 Microsoft Visual C++ LCG

In Figure 6.9 we report the results of the DECT Suite applied to sequences of different length produced by the Microsoft Visual C++ LCG, see Table A.5. In detail, a single $100,000,000$-bit long file (named *MS-LCG-100Mb*) was initially generated, from which three files (named *MS-LCG-16*, *MS-LCG-19* and *MS-LCG-22*) were derived by truncating the first file to the desired length (2^{16}, 2^{19} and 2^{22}, respectively).

We observe that with $L = 2^{16}$, Figure 6.9a, the DECT Suite does not reveal any statistical anomalies (a single $W1$ tag which is not meaningful, as discussed in §6.4). However, when we move to $L = 2^{19}$, Figure 6.9b, and even more to $L = 2^{22}$, Figure 6.9c, we gain strong evidence of non-randomness of the analyzed sequence, with the appearance of high-index tags (we recall from §6.4 that a $W < t >$ or $Q < t >$ tag corresponds to a Type I Error probability smaller than 10^{-t}).

The above observations imply that, at least for certain classes of LCGs, the DECT Suite can be used as a distinguisher, that is, an algorithm capable of identifying statistical irregularities in LCG sequences which are not present in ideal random data[22]. To support this claim, we conducted the following experiment. First we generated 100 sequences, consisting of 50 produced with AES-GCM (and, thus, assumed to behave as ideally random) and 50 with the Microsoft Visual C++ LCG[23]. Then, we analyzed all of them using the DECT Suite, having (arbitrarily) established

[21] We recall that it is well-known that LCG-generated sequences have good but non optimal statistical behaviour, see §A.8 and §6.5.3.3, hence, it makes sense that a statistical test is able to detect some anomalies.

[22] At the moment it is not clear to us on which classes of LCGs the test is effective. Based on some experiments, we can hypothesize that, in its current form, the test works well only when the modulus is a power of 2 (which, by the way, is by far the most common practical case). It is reasonable that, properly modifying the block length (currently set to 2^{λ}, see §6.1), it can work with other values of the modulus. However, this is an aspect that needs to be further investigated in future work.

[23] For completeness, we mention that the AES-GCM sequences were generated from 50 distinct (key, nonce) pairs, and similarly, the LCG sequences were generated from 50 distinct seeds. For further details on the mechanisms used and the applied methodology, refer to §A.7, §A.8 and §6.5.1.

that the DECT Suite fails[24] when it determines at least one tag with index greater than or equal to 6 among all the W and Q values[25]. We verified that the DECT Suite failed with all and only the sequences produced with the LCG (while, consequently, it succeeded with all and only the sequences produced with AES-GCM), thus giving evidence of the effectiveness of the DECT Suite as LCG distinguisher.

We also analysed the above-mentioned file *MS-LCG-100Mb* with the NIST-STS suite. In Figure 6.10 we report an extract of the corresponding output. The suite was configured to treat the $100,000,000$-bit long file as a set of 100 sequences, each $1,000,000$-bit long, according to the recommendations of [11]. Here we do not detail the format of the output produced by the NIST-STS tests (see again [11], §5.7). We just mention that, according to [11], §4.2.1, in our setting the NIST procedure requires that at least 96 over 100 sequences are successful for all the tests but the Random Excursion and the Random Excursion Variant tests, which, instead require a 56 over 58 success rate. Column "Proportion" of Figure 6.10 clearly shows that the considered file *MS-LCG-100Mb* passes the NIST-STS suite analysis.

Comparing Figures 6.9 and 6.10, we see that the DECT Suite detects the statistical anomalies of the LCG-generated sequence based on the observation of the 2^{19}-bit long file *MS-LCG-19*, while the NIST-STS suite does not catch any irregularities on the much longer $100,000,000$-bit (approximately $2^{26.58}$-bit) file *MS-LCG-100Mb*. Increasing the length of the sequence analyzed by the NIST-STS suite doesn't seem to yield significant improvements and, furthermore, it quickly becomes computationally intractable (in addition to reporting computation errors). Therefore, based on the conducted experiments, it seems that the DECT Suite performs (significantly) better than the NIST-STS suite on the considered LCG.

6.6.2 Larger LCGs

To complete the analysis, we applied the DECT Suite to sequences produced by LCGs with parameters significantly larger than the Microsoft LCG of §6.6.1. We

[24] Here by "failure" we mean that the application of the suite to a given sequence leads to conclude that the sequence does not support the null hypothesis. Conversely, by"success" we mean that it does, that is, the suite does not detect a statistical anomaly in the generation process.

[25] Strictly speaking, the probability of encountering such an index, under the null hypothesis of uniformity, slightly increases with the length of the analysed sequence, since the number of λ values considered (and thus of DECT-Q Tests and DECT-Q Tests) increases as well. However, since the choice of index 6 as cut-off p-value is subjective, we don't care about this minor uncertainty.

```
File 'MS-LCG-16' of size 2^16 bits

 λ   |======== W =========|======== Q =========|
     |                    |                    |
 7   |       0.1478       |       0.5433       |
 8   |       0.0981 W1    |       0.6407       |
 9   |       0.5893       |       0.5269       |
10   |       0.4786       |       0.6898       |
11   |       0.7424       |       0.2018       |
```

(a) $L = 2^{16}$

```
File 'MS-LCG-19' of size 2^19 bits

 λ   |======== W =========|======== Q =========|
     |                    |                    |
 7   |       0.9791       |       0.0757 Q1    |
 8   |       0.1433       |       0.0944 Q1    |
 9   |       0.7640       |       0.5951       |
10   |       0.0455 W1    |       0.9941       |
11   |       0.0036 W2    |       0.9999       |
12   |       0.0259 W1    |       0.9625       |
13   |       0.0002 W3    |       0.0678 Q1    |
14   |       0.0000 W9    |       0.0010 Q3    |
```

(b) $L = 2^{19}$

```
File 'MS-LCG-22' of size 2^22 bits

 λ   |======== W =========|======== Q =========|
     |                    |                    |
 7   |       0.0157 W1    |       0.8674       |
 8   |       0.7705       |       0.3343       |
 9   |       0.9034       |       0.0237 Q1    |
10   |       0.0008 W3    |       0.0002 Q3    |
11   |       0.0000 W9    |       0.0000 Q7    |
12   |       0.0000 W9    |       0.0000 Q9    |
13   |       0.0000 W9    |       0.0000 Q9    |
14   |       0.0000 W9    |       0.0000 Q9    |
15   |       0.0000 W9    |       0.0000 Q9    |
16   |       0.0000 W9    |       0.0000 Q9    |
17   |       0.0000 W9    |       0.0000 Q7    |
```

(c) $L = 2^{22}$

Fig. 6.9 DECT Suite on LCG, $m = 2^{31}$

```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <MS-LCG-100Mb>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE     PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
 10   9   7   7  11   7  11   9  16  13  0.574903      98/100     Frequency
 11   7   7   7   8  13  12  22   8   5  0.009535     100/100     BlockFrequency
 15   4   8   7  13   8  12  11  11  11  0.401199      98/100     CumulativeSums
  7  15   9   9   9   7  14   9  15   6  0.319084      98/100     CumulativeSums
 15   9  16  10  10  10  10   3  11   6  0.171867      99/100     Runs
  5   7  11  12   8  16   9  13  10   9  0.437274      99/100     LongestRun
  9  10  11  11  15   9   7   9  12   7  0.816537      97/100     Rank
 15  14   9   7  13   7  14   5   8   8  0.224821     100/100     FFT
 10   9  11   9   9  11   9   8  11  13  0.991468      98/100     NonOverlappingTemplate
 12  13   7   8   8  13   8  11  11   9  0.867692      99/100     NonOverlappingTemplate
 12  17   9   7   6   8   6  11  13  11  0.275709      98/100     NonOverlappingTemplate
 10   8  17  13  12   5  11   3   7  14  0.055361      99/100     NonOverlappingTemplate

                                         [...]

  3  11  14  13   7  11  11  11  14   5  0.171867     100/100     NonOverlappingTemplate
  6  13   7  10  15  13  14  10   6   6  0.236810     100/100     NonOverlappingTemplate
 10   8   8   7  12  10  12  11  12  10  0.964295     100/100     NonOverlappingTemplate
 13   9   6   9  11   7   4   9  18  14  0.080519      99/100     OverlappingTemplate
 12  12   9   9  10   6  10  15  11   6  0.657933     100/100     Universal
 15   9  13  11   8   7  13   7   9   8  0.616305     100/100     ApproximateEntropy
 10   5   5   7   5   5   5   4   7   5  0.657933      56/58      RandomExcursions
  5   5   7   8   7   3   6   5   5   7  0.816537      58/58      RandomExcursions
  7   4  11   3   4   3   4   9   9   4  0.051942      57/58      RandomExcursions
  8   4  11   4   8   7   3   9   2   2  0.020548      58/58      RandomExcursions
  6   6   4   7   4   6   9   5   4   7  0.739918      57/58      RandomExcursions
  5   8   4   5   3   4   7   9   3  10  0.171867      58/58      RandomExcursions
  6   5   3   6   5   5   9   5   4  10  0.383827      58/58      RandomExcursions
  6   4   4   4   7   7   6  10   3   7  0.419021      56/58      RandomExcursions
  6   8   7   5   6   5   4   3   6   8  0.739918      58/58      RandomExcursionsVariant
  8   5   7   2   6   7   4   6   7   6  0.657933      58/58      RandomExcursionsVariant
  6   5  11   3   1   6   6   6   9   5  0.085587      57/58      RandomExcursionsVariant
  8   3   5   8   3   5   5   8   5   8  0.455937      55/58      RandomExcursionsVariant
  8   6   2   5   8   5   5   6   5   8  0.574903      57/58      RandomExcursionsVariant
  7   3   6   8   9   5   5   4   6   5  0.616305      57/58      RandomExcursionsVariant
  8   4   4   7  11   1   3   9   9   2  0.007694      57/58      RandomExcursionsVariant
  6   8   4   5   6   7   4   6   5   7  0.883171      57/58      RandomExcursionsVariant
  4   5   9   5   5   6   5   6   6   7  0.851383      57/58      RandomExcursionsVariant
  8   6   6   3   9   5   6   8   4   3  0.419021      57/58      RandomExcursionsVariant
  7   2   6  11   4   6   6   5   7   4  0.236810      57/58      RandomExcursionsVariant
  7   5   2   7   5  13   6   4   8   1  0.010237      57/58      RandomExcursionsVariant
  7   4   5  10   2   9   4   6   4   7  0.191687      57/58      RandomExcursionsVariant
  3   5   4   6  12   9   5   5   4   5  0.108791      58/58      RandomExcursionsVariant
  4   3   4   6   7   8  11   6   7   2  0.122325      58/58      RandomExcursionsVariant
  4   3   4   6  10   3   8   6  10   4  0.108791      57/58      RandomExcursionsVariant
  5   3   9   3   6   7   7   4   4  10  0.213309      58/58      RandomExcursionsVariant
  6   7   5   6   3   6   5   9   7   4  0.699313      58/58      RandomExcursionsVariant
 15   8   8  10   6   7  13  15   8  10  0.383827      99/100     Serial
  7  16   8   8   7   8  12  12  14   8  0.401199      97/100     Serial
 13  11  10  10   9  11  11  10  10   5  0.924076     100/100     LinearComplexity


- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
The minimum pass rate for each statistical test with the exception of the
random excursion (variant) test is approximately = 96 for a
sample size = 100 binary sequences.

The minimum pass rate for the random excursion (variant) test
is approximately = 55 for a sample size = 58 binary sequences.

For further guidelines construct a probability table using the MAPLE program
provided in the addendum section of the documentation.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Fig. 6.10 NIST-STS on LCG, $m = 2^{31}$

considered several LCGs with increasing module size (power of 2) and randomly chosen (multiplicative and additive) constants, satisfying the constraints expressed in §A.8 to guarantee the maximum period.

Moreover, in addition to the DECT Suite, we selected other suites in order to compare their performances with those of the DECT Suite on the extended set of LCGs. In particular, among those mentioned in §5.1, we excluded Knuth's suite due to its current limited usage and Diehard since it is included in Dieharder. Instead, we opted to retain PractRand, gjrand, TestU01, and Dieharder as they are widely used or strongly emerging tools in the cryptographic community.

With regard to the NIST-STS suite, we were not able to include it in our broader analysis. As expected, we observed that in general the use of larger LCG modulus values results in a substantial increase in file sizes for the input sequences in order to make biases detectable. Unfortunately this rendered the NIST-STS suite impractical due to the associated computational complexity and the increased probability of calculation errors. Thus, we had to exclude the NIST-STS from analysis involving sequences significantly longer than those considered in §6.6.1, that is, $100,000,000$ bits.

The comparison between the suites was conducted as follows. For each suite and each LCG, a sequence was generated by iteratively evolving the generator internal state, which was initially seeded with an arbitrary value. At each step, the block made of the 16 most significant bits of the resulting internal state were extracted and appended to the previous block, gradually extending the length of the sequence. During the generation process, at each doubling of the sequence length, the suite was applied to the obtained sequence. If a statistical anomaly was detected, according to the (arbitrary) criterion that at least one p-value produced by the suite on the sequence was less than 10^{-6}, the generation was stopped and the sequence was deemed to be non-random. Alternatively, the process ended when the length of the sequence reached 8Tb without any statistical bias being detected, and the sequence was considered random.

We were interested in observing whether the suite was capable of detecting the statistical anomaly of the selected LCG and, if so, how long the considered sequence needed to be. In Table 6.10 a summary of the obtained results is provided. Among the generator parameters, only the modulus is reported (first column) because in our experiments we observed that results are largely independent of the multiplicative and

Modulus size	Minimum size for failure (base-2 logarithm)				
	DECT Suite	**PractRand**	**gjrand**	**TestU01**	**Dieharder**
32	18	22	27	30	41
40	21	28	27	30	not found
48	24	28	27	30	not found
56	25	30	27	30	not found
64	26	33	30	40	not found
72	28	35	33	not found	not found
80	30	38	37	not found	not found
96	31	43	not found	not found	not found
112	33	not found	not found	not found	not found

Table 6.10 Test suites comparison on LCGs

additive constants (when properly chosen, according to the conditions for maximum period given in §A.8)[26]. Then, in the following columns, we report the results of DECT Suite, PractRand, gjrand, TestU01 and Dieharder. For each combination of suite and modulus, the minimum length of the sequence for which the suite was able to detect a statistical irregularity is reported, expressed for simplicity through its base-2 logarithm[27]. The label "not found" in red, instead, indicates that the suite failed to recognize the imperfect randomness of the sequence (within a sequence length limit, set to 2^{43} bits), despite the sequence being produced by a non-ideally random generator (namely, a LCG).

From the preliminary results reported in Table 6.10 it appears that, in the specific case of the LCGs, the DECT Suite performs significantly better than the other considered suites, that is, it needs (much) shorter sequences to detect the statistical anomaly of the random generator. The exact reasons behind the reported results are currently under examination and, in fact, a possible future direction of work could be to gain a better understanding of the relation between the DECT Suite and the LCGs structure.

[26]For each set of parameters, we conducted multiple experiments with different seeds. Obviously, results vary from experiment to experiment, but we observed that the average values (reported here) do not seem to be significantly affected by the choice of parameters, for a given modulus.

[27]For the sake of completeness, we note that gjrand, TestU01 and Dieharder can only perform analyses on pre-defined sizes of input sequences. Specifically, gjrand handles only the following lengths: 27, 30, 33, 37, 40, 42, while TestU01 exclusively allows the following lengths: 30, 40, 43 and, finally, Dieharder only 41 (lengths are still expressed in bits through their base-2 logarithm).

6.7 DECT Suite open points

While the definition of the suite is complete (§6.3), appears well validated (§6.5), and also yields concrete results (§6.6), several points have emerged during the development of the chapter that would benefit from further exploration. A short list of possible objectives for future works follows.

- To fully understand why the DECT Suite is (so) effective on the LCGs we have tested;
- to adapt the DECT Suite to LCGs whose modulus is not power of 2;
- to investigate the relation among the outputs corresponding to different values of λ for the DECT-W Test and the DECT-Q Test;
- to analyse more in depth the behaviour of the DECT-W Test and the DECT-Q Test on the border cases, both of λ and of the p-value, checking if the employed approximations work as accurately as expected;
- to increment the maximum size of Table 6.10. At the moment, we had to stop at 112 as maximum modulus size, due to the computational complexity required, since the current DECT Suite implementation is not able to deal with sequences longer than 2^{34} bits. However, it is conceivable that by working on the implementation of the suite, we may achieve better results (the first target being obviously 128, which seems to be within reach observing Table 6.10);
- to study whether, for a fixed modulus size, the effectiveness of the DECT Suite can be influenced by the choice of the parameters of the LCG. Preliminary experiments seem to reject this hypothesis, in that even the parameters proposed in [56], which show higher quality in certain regards, do not appear to yield different results compared to randomly generated ones. However, a more in-depth analysis could certainly provide different insights;
- to apply the DECT Suite to the Lehmer generators, a specific and interesting case of LCG introduced in §A.8.

Chapter 7

Conclusions

The purpose of this dissertation is to further investigate the use of hypothesis testing applied to the validation of random number generators, with a specific focus on cryptography.

The field of cryptography is indeed highly demanding in terms of the quality of randomness involved, as it directly impacts the generation of encryption keys and other relevant parameters. It is no coincidence that a significant number of cryptographic system failures can be attributed to poor utilization of randomness. In Chapter 1, we provide a brief list of well-known randomness-related failure examples along with other fundamental concepts, like TRNGs and PRNGs.

It is therefore essential to have reliable tools for verifying the quality of random number generators. One commonly used tool is hypothesis testing and the related concept of p-value, which allows for an assessment of a certain null hypothesis about the random generator (typically assuming that the produced sequences are uniformly distributed and independent). Usually, the probability distribution underlying the null hypothesis is continuous or assumed to be well approximated by a continuous distribution (often chosen from a limited set of common and well-studied distributions), making the analysis more manageable. These tools are described in Chapter 2.

From Chapter 3 to Chapter 6, the original contribution of this dissertation is presented, approaching the topic from different yet complementary perspectives: a detailed analysis of the distribution of p-values in the discrete case (Chapter 3), an abstract generalization of the concept of hypothesis testing (Chapter 4), an examina-

tion of the NIST standard test suite (Chapter 5), and finally, a concrete proposal of a hypothesis test and validation methodology (Chapter 6).

In particular, in Chapter 3 it is initially observed that in the case of random number generators, the null hypothesis is based on an underlying discrete probability distribution. Consequently, the common approximations with continuous distributions, while highly effective in many other contexts, are here not entirely precise. Throughout the chapter, we derive general results on the distribution of p-values for a given test, based on the underlying data distribution (i.e., the probability distribution of the input samples to the test). We investigate three cases, progressively refining the definitions: the first case considers no constraint on the data distribution, the second case assumes an arbitrary but fixed data distribution, and finally, the third case focuses on a uniform data distribution. The latter case holds particular significance for us as it encompasses random number generators that follow a uniform probability distribution. Based on this analysis, we also explore the possibility (suggested by NIST) of utilizing the alleged uniformity of the p-value distribution for an arbitrary test to construct an analysis procedure for multiple samples. In this regard, we note that the mentioned uniformity holds in the continuous case but only approximately in the discrete case. We present counterexamples and derive conditions for (cautiously) applying it to the discrete case.

In Chapter 4, we construct a more abstract and general model of hypothesis testing compared to its conventional description. Essentially, we identify a hypothesis test with a partition of the sample space into two subsets: one corresponding to all the samples that support the null hypothesis and the other corresponding to all the samples that reject it. Subsequently, we characterize various properties of the described model and we elaborate on the interpretation of concretely implementable tests as specific instances of the general framework.

Later, building upon the results of the previous chapters, in Chapter 5 we thoroughly analyze specific aspects of the statistical test suite proposed by NIST, demonstrating that certain strategies, suggested for analyzing sequences produced by random number generators, are not entirely accurate. This is primarily due to the reliance on assumptions regarding the continuity of the data distribution, which are only approximately valid in the discrete setting.

Finally, in Chapter 6, we construct a simple hypothesis test suite for random generators, named DECT Suite, providing a detailed analysis of the resulting distribu-

tions of p-values and establishing a methodology for verifying the correct definition and implementation of the suite. Additionally, we observe that the DECT Suite, despite its simplicity, proves to be surprisingly effective against a well-known and widely used class of PRNG, namely the LCGs (not recommended for cryptographic purposes). Remarkably, in this specific case, the DECT Suite appears to be even more effective than the most used test suites.

Appendix A

Useful concepts

This appendix contains basic notions about some concepts used throughout the previous chapters, with the only purpose to provide a quick high-level reference to the relevant topics. In particular, the addressed elements are from the fields of Probability Theory (from §A.1 to §A.5), Information Theory (§A.6) and Cryptography (§A.7 and §A.8).

A.1 Random variables

Given a random experiment, let S be its sample space, that is, the set of all possible outcomes. A random variable X is a function mapping the sample space to a real number.

$$X : S \to \mathbb{R}$$

A random variable can be *discrete*, if $X(S)$ (that is, the set of possible values of X) is finite or countably infinite, or *continuous*, if it is uncountably infinite.

Given a random variable denoted by X, we typically indicate by x its *realization*, that is the observed value of X in a specific experiment.

See [12] and [57] for a more in-depth introduction to the topic.

A.2 Probability distribution

A probability distribution is associated to the random variable X. It describes the set of all possible values of the variable and their associated probabilities, providing a complete representation of how likely each value is to occur. A probability distribution is said *discrete* or *continuous* according to the nature of the associated random variable.

For a detailed and comprehensive treatment of the topics of this section (§A.2), see [12] and [57].

A.2.1 Cumulative Distribution Function (CDF)

A way to describe the distribution of a random variable X, both for discrete and continuous case, is through its CDF, defining the probability that it takes values less than or equal to any given value.

Definition 15. *Given a random variable X, its CDF $F_X(x)$ is defined as*

$$F_X(x) = Pr(X \leq x)$$

A.2.2 Probability Mass Function (PMF)

Equivalently, in the discrete case, the probability distribution of X can be given by its PMF, defining the probabilities of occurrence of all the possible outcomes.

Definition 16. *Given a discrete random variable X, whose set of possible outcomes is $X(S) = \{x_1, x_2, \ldots\}$, its PMF f_X is defined for a given x_i as*

$$f_X(x_i) = Pr(X = x_i)$$

that is, the probability that the random variable X takes the value x_i.

CDF and PMF for a discrete random variable are linked by the following relation:

$$F_X(x) = \sum_{x_i \leq x} f_X(x_i)$$

A.2.3 Probability Density Function (PDF)

For the continuous case, however, a different approach is required, since continuous random variables are not defined at specific values. Instead, they are defined over intervals of values (and the probability of observing a specific value is 0), and can be described by their probability density function.

Definition 17. *Given a continuous random variable X, its PDF $f_X(x)$ is defined as*

$$f_X(x) = \frac{d}{dx} F_X(x)$$

The above relation can be inverted as

$$F_X(x) = \int_{-\infty}^{x} f_x(t)dt$$

Thus, given two arbitrary a and b, we have the following useful relation:

$$Pr(a \leq X \leq b) = \int_{a}^{b} f_X(x)dx$$

We observe that the same notation f_X is used both for the PMF and the PDF, because the meaning of the two functions is similar. However no confusion arises, because they refer to two distinct contexts (discrete and continuous variables).

A.3 Statistical measures

Many statistical measures can be associated with a random variable. The three most commonly used are mean, variance and standard deviation. All of them play a crucial role in characterizing probability distributions and understanding the behavior of random variables.

For a detailed introduction to the concepts of this section (§A.3) refer to [12] and [57].

A.3.1 Mean

The mean (or expected) value of a random variable X, denoted by μ_X, serves as a measure of its central tendency, indicating where the average value of the variable is located. If X is a discrete random variable, then

$$\mu_X = E(X) = \sum_{x_i \in X(S)} f_X(x_i) x_i$$

If X is a continuous random variable, then $X(S)$ coincides with the real axis and

$$\mu_X = E(X) = \int_{-\infty}^{\infty} x f_X(x) dx$$

A.3.2 Variance

The variance of a random variable X, denoted by σ_X^2, quantifies the spread or dispersion of a random variable's probability distribution, providing information about how the values of the variable deviate from their mean or expected value. variance is defined as

$$Var(X) = \sigma_X^2 = E((X - \mu_X)^2)$$

In the discrete setting this means

$$Var(X) = \sum_{x_i \in X(S)} (x_i - \mu_X)^2 f_X(x_i)$$

while in the continuous setting we have

$$Var(X) = \int_{-\infty}^{\infty} (x - \mu_x)^2 f(x) dx$$

A.3.3 Standard Deviation

The standard deviation of a random variable X, denoted by σ_X, is calculated as the square root of the variance.

$$\sigma_X = \sqrt{Var(X)}$$

It provides a measure of the average distance between each value of X and the mean μ_X, indicating how much the values globally deviate from the average. With respect to the variance, the standard deviation allows for a more intuitive interpretation of the spread of a distribution, as it is expressed in the same units as the random variable itself, resulting more user-friendly and easier to interpret.

A.4 Relevant distributions

Here we briefly consider the probability distributions referred to in the previous chapters. In fact, all of them belong to more general distributions families, which can be specified through the use of one or more distribution parameters, as shown below. For each family of distributions, we report the distribution parameters, the support (that is, the set of all possible values for which the probability of occurrence is positive) and the most relevant statistical measures, namely mean, variance, PMF or PDF, and CDF.

For a thorough analysis of the distributions considered in this section (§A.4) see [12] and [57] and, specifically for §A.4.4, also [55].

A.4.1 Discrete Uniform Distribution

The discrete Uniform Distribution U describes the setting where a finite number of values from a given interval ($[a,b]$) can be observed with equal probability. It is typically indicated as $U(a,b)$.

• Parameters: $a, b \in \mathbb{N}$, with $a \leq b$; $n = b - a + 1$; • support: $k \in [a, b] \subset \mathbb{N}$;
• mean: $\frac{a+b}{2}$; • variance: $\frac{n^2-1}{12}$; • PMF: $\frac{1}{n}$, for any k; • CDF: $\frac{k-a+1}{n}$, for any k.

Table A.1 Discrete Uniform Distribution

A.4.2 Binomial Distribution

The binomial distribution is a discrete probability distribution that models the number of successes in a fixed number (n) of independent Bernoulli trials, where each trial has the same probability of success (p). It is typically indicated as $B(n, p)$.

• Parameters: $n \in \mathbb{N}, p \in [0,1] \subseteq \mathbb{R}$; • auxiliary parameter: $q = 1 - p$; • support: $k \in \{0,1,\ldots,n\}$ = number of successes;
• mean: np; • variance: npq; • PMF: $\binom{n}{k} p^k q^{n-k}$; • CDF: $I_q(n-k, 1+k)$, where I_q is the regularized beta function.

Table A.2 Binomial Distribution

A.4.3 Normal Distribution

The normal distribution, also known as the Gaussian distribution or bell curve, is a continuous probability distribution that is symmetric and characterized by its mean (μ) and variance (σ^2). It is typically indicated as $\mathcal{N}(\mu, \sigma^2)$.

• Parameters: $\mu \in \mathbb{R}$, $\sigma^2 \in \mathbb{R}^+$; • support: $x \in \mathbb{R}$;
• mean: μ; • variance: σ^2; • PDF: $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$; • CDF: $\frac{1}{2}\left[1 + erf\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right]$, where erf is the error function.

Table A.3 Normal Distribution

Among many useful facts about the Normal Distribution, the following two are especially relevant and have been used through the dissertation.

A.4.3.1 Normal Distribution approximation to the Binomial Distribution

Under certain conditions, the Normal Distribution provides a good approximation to the Binomial Distribution. More precisely, given $B(n,p)$, with $q = 1-p$, the following approximation holds:

$$B(n,p) \sim \mathcal{N}(np, npq)$$

provided that

$$np \geq 10, nq \geq 10 \tag{A.1}$$

as a rule of thumb[1].

A.4.3.2 Standard Normal Transformation

The Standard Normal Transformation converts a normally distributed random variable to the Normal Standard Distribution, which is easier to manage, since it has well-known properties and established tables.

Given a normally distributed random variable $X \sim \mathcal{N}(\mu, \sigma^2)$, the Standard Normal Transformation determines a new variable Z as follows:

$$Z = \frac{X - \mu}{\sigma}$$

Z is centered at $\mu_Z = 0$ and has variance $\sigma_Z^2 = 1$, that is, $Z \sim \mathcal{N}(0,1)$.

A.4.4 Gumbel Distribution

The Gumbel distribution, also known as the Type I Extreme Value distribution, is a continuous probability distribution commonly used, under certain conditions, to model the maximum value in a set of random variables. The Gumbel distribution has

[1] In literature it is often suggested the alternative condition $np \geq 5, nq \geq 5$. We prefer the form given in Equation (A.1), as more conservative.

an asymmetric, right-skewed, single-peaked shape with a long tail extending to the right. It is characterized by two parameters, namely the location (μ) and the scale (β), and is typically indicated as $G(\mu, \beta)$.

• Parameters: $\mu \in \mathbb{R}$, $\beta \in \mathbb{R}^+$; • auxiliary constant: $\gamma \approx 0.57721$, the Euler-Mascheroni constant; • support: $x \in \mathbb{R}$;
• mean: $\mu + \beta\gamma$; • variance: $\frac{\pi^2\beta^2}{6}$; • PDF: $\frac{1}{\beta} e^{-(\frac{x-\mu}{\beta} + e^{-\frac{x-\mu}{\beta}})}$; • CDF: $e^{-e^{-\frac{x-\mu}{\beta}}}$.

Table A.4 Gumbel Distribution

In [55], pp. $3-9$, it is proven, in particular, that the Gumbel distribution is a good approximation for the maximum of a set of n independent variables with identical CDF F. In this setting, it is also estimated the value of the distribution parameters, as:

$$\mu = F^{-1}\left(1 - \frac{1}{n}\right), \beta = F^{-1}\left(1 - \frac{1}{ne}\right) - \mu$$

A.5 χ^2 Goodness of Fit Test

A goodness-of-fit test [25], [26], [27] is an hypothesis test used to determine how well a sample of observed data fits a specific theoretical distribution. It assesses whether the observed data significantly deviate from the expected distribution. One commonly used goodness-of-fit test is the chi-square test.

The chi-square (χ^2) test compares the observed frequencies in a finite number of different categories with the expected frequencies based on a theoretical distribution. It produces a p-value which measures the discrepancy between the observed and expected frequencies.

The test works as follows. Given k possible outcomes (categories) and the corresponding probabilities $p_i, i = 1,2,\ldots,k$, hypothesized according to the null hypothesis, the vector $(E_i = np_i, i = 1,2,\ldots,k)$ represents the expected counts for n observed samples. Moreover, let $(O_i, i = 1,2,\ldots,k)$ the vector of the observed values. Then the χ^2 statistics can be computed as follows:

$$\chi^2 = \sum_{i=1}^{k} \frac{(E_i - O_i)^2}{E_i}$$

Then a p-value can be associated to the obtained χ^2 statistics. The p-value is a function of the χ^2 value and of another parameter r (the number of degrees of freedom), which in our setting is simply $r = k - 1$. Here it is not of interest to delve into the details, which can be found for instance in [12], p. 760, and on most statistical books. We just mention that the computation of the p-value is available on most specialized software tools and on various online resources.

A commonly accepted criterion for the application of a χ^2 test is that each expected value (E_i in Equation (A.5)) is at least 5:

$$E_i \geq 5, i = 1,2,\ldots,k$$

A.6 Entropy

In the context of information theory, entropy is a fundamental concept that measures the uncertainty or information content of a random variable or a probability distribution. Measuring the uncertainty (that is, the randomness) is a quite subtle task, due to the elusive nature of the randomness. Thus, not surprisingly, many different definitions of entropy are in use, the most commonly used being probably the Shannon Entropy, which is typically defined for discrete random variables[2].

[2]Extension of the Shannon Entropy to the continuous case is possible, but it requires appropriate mathematical and interpretive treatment and is not useful in this context.

Definition 18. *Given a discrete random variable X, the Shannon Entropy $H(X)$ is defined as:*

$$H(X) = -\sum_{x_i \in X(S)} f_X(x_i) \log_2 f_X(x_i)$$

where f_X is the PMF defined on the set of possible outcomes $\{x_1, x_2, \ldots\}$.

Refer to [58] and [59] for an in-depth analysis of the concept of entropy.

A.7 AES

AES is a widely used symmetric encryption algorithm that provides a high level of security for protecting sensitive information, specified by NIST in [60]. More precisely, AES is a block cipher, that is, it transforms plaintext data into ciphertext data, operating on fixed-size (128 bits) blocks of data and employing a (secret) key (128, 192 or 256 bits) to perform the encryption and decryption processes. Despite being block-oriented, there are many ways a block cipher can be converted into a PRNG (see §1.3.2), able to produce (virtually) endless pseudo-random sequences. Among them, a commonly used method, mentioned in the previous chapters, is the Galois Counter Mode (GCM) defined in [61]. When used in combination with AES, it is referred to as AES-GCM. Depending on the key[3] and on another random value (referred to as nonce), AES-GCM produces sequences (also known as *keystreams*) which are believed to be (for any practical purpose) indistinguishable from random data. We observe that these sequences do not depend on the plaintext data but only on the key and the nonce[4]. Moreover, AES-GCM ensures that, given two different (key, nonce) pairs, the corresponding sequences are considered (for any practical purpose) independent of each other.

A good introduction to cryptography, and in particular to AES, can be found in [62] and [31].

[3] We note that for cryptographic application it is assumed that the key is kept secret. However, when AES is used as a building block to produce a (strong but not crypto-oriented) PRNG, this requirement can be relaxed.

[4] In practice, when AES-GCM is used to encrypt/decrypt data, the output keystream is bitwise xor-ed with the plaintext to produce the ciphertext. Vice versa, in order to decrypt the ciphertext, it is bitwise xor-ed with the keystream to produce back the plaintext.

A.8 Linear Congruential Generators

LCGs are class of PRNG which produce random-looking sequences of numbers based on a linear recurrence relation. More precisely, given an initial seed, they repeatedly multiply a previous number by a constant, add another constant, divide by a modulus and take (some selected bits of) the remainder as output.

Formally, a LCG is then defined by three integer constants:

- a modulus m, with $0 < m$;
- a multiplicative constant a, with $0 < a < m$;
- an additive constant c, with $0 \leq c < m$;

Given a seed X_0, the internal state X_i evolves according to the following linear recurrence:

$$X_{i+1} = (aX_i + c) \mod m$$

LCGs are widely used in scenarios where efficiency and simplicity are prioritized over advanced statistical properties (for instance, they are often used in Monte Carlo simulations [33], [34]). In fact, they exhibit some statistical weaknesses that make them unsuitable for direct use in cryptography, but they can be found as components in more complex cryptographic systems.

Careful consideration should then be given to selecting appropriate constants values, since they can impact on the quality of the generated sequence in terms of randomness and statistical properties. In particular, it is well known that the maximum period (m) is obtained for $c \neq 0$ if and only if the following three conditions hold:

- m and c are coprime;
- all prime factors of m divide $a - 1$;
- if 4 divides m then 4 divides $a - 1$.

LCGs have been proposed with many set of constants. Three among the most commonly used are given in Table A.5, where the fourth column contains the number

of (most significant) bits of the internal state taken as output at each step. All the listed LCGs satisfy the above requirements for maximum period (2^{31}).

Generator	m (modulus)	a (multiplier)	c (increment)	n (output size)
Microsoft Visual C++	2^{31}	214013	2531011	15
Borland C	2^{31}	22695477	1	15 or 31
ANSI	2^{31}	1103515245	12345	15

Table A.5 Some common LCGs

In the specific case where the additive constant is zero, $c = 0$, the resulting generator is called Lehmer generator [63], also known as Park–Miller generator [64].

The Lehmer generator is attractive because simpler to implement, but it requires more attention in choosing the parameters. Specifically, the maximum period (m-1) is achieved when m is prime, a is a primitive root modulo m, and finally, the seed X_0 is coprime with m.

References

[1] Ian Goldberg and David Wagner. Randomness and the Netscape browser. 1996. URL https://people.eecs.berkeley.edu/~daw/papers/ddj-netscape.html.

[2] Meltem Sönmez Turan, Elaine Barker, John Kelsey, Kerry A McKay, Mary L Baish, and Mike Boyle. Debian Security Advisory. DSA-1571-1 openssl – predictable random number generator. Technical report, National Institute of Standards and Technology, 2008. URL https://www.debian.org/security/2008/dsa-1571.

[3] Arjen K. Lenstra, James P. Hughes, Maxime Augier, Joppe W. Bos, Thorsten Kleinjung, and Christophe Wachter. Ron was wrong, Whit is right, 2012. URL https://eprint.iacr.org/2012/064.pdf.

[4] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[5] X.509 : Information technology - open systems interconnection - the directory: Public-key and attribute certificate frameworks. https://www.itu.int/rec/T-REC-X.509. Accessed: 2023-08-07.

[6] Peter Gutmann. Engineering security, 2014. URL https://www.cs.auckland.ac.nz/~pgut001/pubs/book.pdf. Accessed: 2023-08-07.

[7] Philip Zimmermann. Where to get pgp. URL https://philzimmermann.com/EN/findpgp/. Accessed: 2023-08-07.

[8] Mario Stipčević and Çetin Koç. *True Random Number Generators*, pages 275–315. 11 2014. ISBN 978-3-319-10682-3. doi: 10.1007/978-3-319-10683-0_12. URL https://www.researchgate.net/publication/299824248_True_Random_Number_Generators.

[9] L.M. Surhone, M.T. Timpledon, and S.F. Marseken. *Pseudorandom Number Generator: Algorithm, Randomness, Monte Carlo Method, Procedural Generation, Cryptography, Linear Congruential Generator, Lagged Fibonacci Generator, Feedback with Carry Shift Registers*. Betascript Publishing, 2010. ISBN 9786130321383. URL https://books.google.it/books?id=cUsXQwAACAAJ.

[10] Elaine B. Barker and John M. Kelsey. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. Technical report, National Institute of Standards and Technology, 2015. URL https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf.

[11] Meltem Sönmez Turan, Elaine Barker, John Kelsey, Kerry A McKay, Mary L Baish, and Mike Boyle. Recommendation for the Entropy Sources Used for Random Bit Generation. Technical report, National Institute of Standards and Technology, 2018. URL https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf.

[12] M.J. Evans and J.S. Rosenthal. *Probability and Statistics: The Science of Uncertainty*. W. H. Freeman, 2009. ISBN 9781429281270. URL https://books.google.it/books?id=plokAAAAQBAJ.

[13] James Jaccard and Michael A Becker. *Statistics for the behavioral sciences*. Cengage Learning, 2021.

[14] Duncan J Murdoch, Yu-Ling Tsai, and James Adcock. *P* -Values are Random Variables. 62(3):242–245, 2008. ISSN 0003-1305, 1537-2731. doi: 10.1198/000313008X332421. URL http://www.tandfonline.com/doi/abs/10.1198/000313008X332421.

[15] Todd A. Kuffner and Stephen G. Walker. Why are *p* -Values Controversial? 73(1):1–3, 2019. ISSN 0003-1305, 1537-2731. doi: 10.1080/00031305.2016.1277161. URL https://www.tandfonline.com/doi/full/10.1080/00031305.2016.1277161.

[16] Hector Zenil. *Randomness through computation: Some answers, more questions (Chapter 3)*. World Scientific, 2011. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=903151.

[17] Lawrence E Bassham III, Andrew L Rukhin, Juan Soto, James R Nechvatal, Miles E Smid, Elaine B Barker, Stefan D Leigh, Mark Levenson, Mark Vangel, David L Banks, et al. *Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications*. National Institute of Standards & Technology, 2010. URL https://csrc.nist.gov/publications/detail/sp/800-22/rev-1a/final.

[18] Craig Blocker, John Conway, Luc Demortier, Joel Heinrich, Tom Junk, Louis Lyons, and Giovanni Punzi. Simple Facts about P -Values. 2006. URL https://hep-physics.rockefeller.edu/~luc/technical_reports/cdf8023_facts_about_p_values.pdf.

[19] Dun Qiu and Jeffrey Remmel. Patterns in words of ordered set partitions. *arXiv preprint arXiv:1804.07087*, 2018. URL https://arxiv.org/abs/1804.07087.

[20] Ronald L Graham, Donald Ervin Knuth, and Oren Patashnik. Concrete mathematics. *Reading, MA*, pages 257–267, 1994. URL https://www.csie.ntu.edu.tw/~r97002/temp/Concrete%20Mathematics%202e.pdf.

[21] Zsófia Kereskényi-Balogh and Gábor Nyul. Fubini numbers and polynomials of graphs. *Mediterranean Journal of Mathematics*, 18:1–10, 2021. URL https://link.springer.com/content/pdf/10.1007/s00009-021-01838-x.pdf.

[22] Neil J. A. Sloane and The OEIS Foundation Inc. The on-line encyclopedia of integer sequences, sequence number A000670. URL https://oeis.org/A000670.

[23] Henry Bottomley and The OEIS Foundation Inc. The on-line encyclopedia of integer sequences, sequence number A000670. URL https://oeis.org/A061095.

[24] Jean-Pierre Barthélémy. An asymptotic equivalent for the number of total preorders on a finite set. *Discrete Mathematics*, 29(3):311–313, 1980. URL https://doi.org/10.1016/0012-365X(80)90159-4.

[25] Ralph B. D'Agostino and Michael A. Stephens. *Goodness-of-fit-techniques*. Routledge, 2017.

[26] Narayanaswamy Balakrishnan, Vassilly Voinov, and Mikhail Stepanovich Nikulin. *Chi-squared goodness of fit tests with applications*. Academic Press, 2013.

[27] William F. Guthrie. *NIST/SEMATECH e-Handbook of Statistical Methods (NIST Handbook 151)*. National Institute of Standards and Technology. doi: 10.18434/M32189. URL https://www.itl.nist.gov/div898/handbook/apr/section2/apr232.htm.

[28] Private communication with Vittorio Bagini and Francesco Stocco, 2023.

[29] Private communication (sipping a beer) with Marco Coppola, 2023.

[30] John Scherk. *Algebra: a computational introduction*. CRC Press, 2000.

[31] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

[32] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and Raphael C.-W. Phan. *The Hash Function BLAKE*. Information Security and Cryptography. Springer Berlin Heidelberg : Imprint: Springer, 1st ed. 2014 edition, 2014. ISBN 978-3-662-44757-4. doi: 10.1007/978-3-662-44757-4. URL https://link.springer.com/book/10.1007/978-3-662-44757-4.

[33] Malvin H Kalos and Paula A Whitlock. *Monte carlo methods*. John Wiley & Sons, 2009.

[34] Mikael Amelin. Monte carlo simulation in engineering, 2013.

[35] Elena Almaraz Luengo and Luis Javier García Villalba. Recommendations on Statistical Randomness Test Batteries for Cryptographic Purposes. 54(4):1–34, 2022. ISSN 0360-0300, 1557-7341. doi: 10.1145/3447773. URL https://dl.acm.org/doi/10.1145/3447773.

[36] Donald Ervin Knuth. *The art of computer programming*, volume 2. Addison-Wesley, 1 edition, 1969. ISBN 978-0-201-89683-1 978-0-201-89684-8 978-0-201-89685-5.

[37] G. Marsaglia. The Marsaglia Random Number CDROM: Including the Diehard Battery of Tests of Randomness. 1995. URL http://ftpmirror.your.org/pub/misc/diehard.

[38] Mohammed M Alani. Testing randomness in ciphertext of block-ciphers using diehard tests. *Int. J. Comput. Sci. Netw. Secur*, 10(4):53–57, 2010. URL https://www.researchgate.net/publication/268414157_Testing_Randomness_in_Ciphertext_of_Block-Ciphers_Using_DieHard_Tests.

[39] R. G. Brown, D. Eddelbuettel, and D. Bauer. Dieharder: A Random Number Test Suite. 2014. URL https://webhome.phy.duke.edu/~rgb/General/dieharder.php.

[40] Marek Sỳs, Lubomír Obrátil, Vashek Matyáš, and Dušan Klinec. A bad day to die hard: Correcting the dieharder battery. *Journal of Cryptology*, 35:1–20, 2022. URL https://link.springer.com/article/10.1007/s00145-021-09414-y.

[41] Pierre L'Ecuyer and Richard Simard. TestU01: A C Library for Empirical Testing of Random Number Generators. 33(4):1–40, 2007. ISSN 0098-3500, 1557-7295. doi: 10.1145/1268776.1268777. URL https://dl.acm.org/doi/10.1145/1268776.1268777.

[42] BD McCullough. A review of testu01, 2006. URL https://www.jstor.org/stable/25146455.

[43] C Doty-Humphrey. Practically random: C++ library of statistical tests for rngs. 2010. URL https://sourceforge.net/projects/pracrand.

[44] Lama Sleem and Raphaël Couturier. Testu01 and practrand: Tools for a randomness evaluation for famous multimedia ciphers. *Multimedia Tools and Applications*, 79:24075–24088, 2020. URL https://link.springer.com/article/10.1007/s11042-020-09108-w.

[45] G. Jones. gjrand random numbers. URL https://gjrand.sourceforge.net/.

[46] Kinga Marton and Alin Suciu. On the interpretation of results from the nist statistical test suite. *Science and Technology*, 18(1):18–32, 2015. URL http://www.romjist.ro/content/pdf/02-msys.pdf.

[47] Song-Ju Kim, Ken Umeno, and Akio Hasegawa. Corrections of the nist statistical test suite for randomness. *arXiv preprint nlin/0401040*, 2004. URL https://arxiv.org/abs/nlin/0401040.

[48] Fabio Pareschi, Riccardo Rovatti, and Gianluca Setti. On statistical tests for randomness included in the nist sp800-22 test suite and based on the binomial distribution. *IEEE Transactions on Information Forensics and Security*, 7(2):491–505, 2012. URL https://ieeexplore.ieee.org/abstract/document/6135498.

[49] Paul Gerhard Hoel et al. Elementary statistics. *Elementary statistics*, pages 313, §6–1, 1960.

[50] Andrew L. Rukhin. *Statistical Testing of Randomness: New and Old Procedures*, pages 33–51. World Scientific, 2011. ISBN 978-981-4327-74-9 978-981-4327-75-6. doi: 10.1142/9789814327756_0003. URL http://www.worldscientific.com/doi/abs/10.1142/9789814327756_0003.

[51] Alessandro Giacchetto. Generatori di numeri aleatori: relazioni tra test di casualità. Technical report, Politecnico di Torino, 2022. URL https://webthesis.biblio.polito.it/24866/1/tesi.pdf.

[52] Fatih Sulak, Muhiddin Uğuz, Onur Kocak, and Ali Doğanaksoy. On the independence of statistical randomness tests included in the nist test suite. *Turkish Journal of Electrical Engineering and Computer Sciences*, 25(5):3673–3683, 2017. URL https://journals.tubitak.gov.tr/elektrik/vol25/iss5/15/.

[53] Elena Almaraz Luengo, Bittor Alaña Olivares, Luis Javier García Villalba, and Julio Hernandez-Castro. Further analysis of the statistical independence of the nist sp 800-22 randomness tests. *Applied Mathematics and Computation*, 459:128222, 2023. URL https://www.sciencedirect.com/science/article/pii/S0096300323003910.

[54] Carmina Georgescu, Emil Simion, Alina-Petrescu Nita, and Antonela Toma. A view on nist randomness tests (in) dependence. In *2017 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pages 1–4. IEEE, 2017. URL https://ieeexplore.ieee.org/abstract/document/8166460.

[55] Samuel Kotz and Saralees Nadarajah. *Extreme value distributions: theory and applications*. World Scientific, 2000.

[56] Guy Steele and Sebastiano Vigna. Computationally easy, spectrally good multipliers for congruential pseudorandom number generators. *Software: Practice and Experience*, 52, 09 2021. doi: 10.1002/spe.3030. URL https://arxiv.org/abs/2001.05304.

[57] Robert B Ash. *Basic probability theory*. Courier Corporation, 2008. URL https://faculty.math.illinois.edu/~r-ash/BPT/BPT.pdf.

[58] Raymond W Yeung. *A first course in information theory*. Springer Science & Business Media, 2002. URL https://link.springer.com/book/10.1007/978-1-4419-8608-5.

[59] Robert B Ash. *Information theory*. Courier Corporation, 2012.

[60] AES Primitives. Advanced encryption standard (aes)(fips-197). 2003. URL https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf.

[61] SP NIST. SP 800-38D. Recommendation for Block Cipher Modes of Operation: Galois. *Counter Mode (GCM) and GMAC, National Institute of Standards and Technology*, 2007. URL https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf.

[62] Christof Paar and Jan Pelzl. *Understanding cryptography: a textbook for students and practitioners*. Springer Science & Business Media, 2009. URL https://link.springer.com/book/10.1007/978-3-642-04101-3.

[63] Derrick H Lehmer. Mathematical models in large-scale computing units. *Ann. Comput. Lab.(Harvard University)*, 26:141–146, 1951.

[64] Stephen K. Park and Keith W. Miller. Random number generators: good ones are hard to find. *Communications of the ACM*, 31(10):1192–1201, 1988. URL https://dl.acm.org/doi/pdf/10.1145/63039.63042.