A NONPARAMETRIC TEST FOR EQUALITY OF DISTRIBUTIONS

by

Florence Dungan

A Thesis Submitted to the Faculty of the

DEPARTMENT OF MATHEMATICS

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

2021

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Master's Committee, we certify that we have read the thesis prepared by: Florence Dungan
titled: A NONPARAMETRIC TEST FOR EQUALITY OF DISTRIBUTIONS

and recommend that it be accepted as fulfilling the thesis requirement for the Master's Degree.

Kevin Lin — Date: May 14, 2021

Hao Helen Zhang — Date: May 19, 2021

Joseph C. Watkins — Date: May 14, 2021

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to the Graduate College.

I hereby certify that I have read this thesis prepared under my direction and recommend that it be accepted as fulfilling the Master's requirement.

Kevin Lin — Date: May 14, 2021
Thesis committee chair
Mathematics

Table of Contents

List of Figures

List of Tables

Abstract

A computationally inexpensive equality test for multivariate distributions is useful in many applications. The goal of this work is to investigate a proposed such test, the projective KS test. It involves projecting the multivariate data onto random lines through the origin and performing one-dimensional Kolmogorov-Smirnov tests on the resulting projections. The projective KS statistic is the maximum of the metrics obtained from those 1D tests.

Two methods of implementation of the projective KS test were developed, one involving sub-sampling in the form of partitioning of the data, and the other using only the whole sample(s). The distribution of the projective KS statistics for both methods was investigated.

For each method, numerical experiments were carried out to calculate the power of the tests for different sample and sub-sample sizes and different numbers of projections, for a range of normal distributions. Power characteristics for the two methods were compared to each other and to the performance of Z-tests. The proposed test (both methods) is less powerful than the Z-tests, but it has the advantage of being nonparametric.

In addition, the projective KS test is potentially useful as a test of random variable independence.

1 Introduction

There is often a need to test if two samples come from the same distribution or if a sample comes from a hypothesized distribution; however, especially in multivariate cases, such a test can get quite expensive from a computational point of view (e.g., Justel *et al.*, 1997).

Sometimes all that is needed is a somewhat quick assessment of whether or not a hypothesis (the null hypothesis) should be rejected: for a one-sample test, that hypothesis states that the sample does come from the hypothesized distribution, while, for a two-sample test, that hypothesis states that the two samples come from the same distribution.

2 Kolmogorov-Smirnov (KS) Test

The Kolmogorov-Smirnov test is a (usually) univariate test based on work by A.N. Kolmogorov (1933, translation available in Shirayev, ed, 1992) and N. Smirnov (1948). It can be used as a one-sample (goodness of fit) test; in that case, the test statistic is the maximum distance between the empirical cumulative distribution function (ECDF) of the sample (see, for example, Knuth, 1998):

$$F_N(x) = \frac{\text{number of } x_1, x_2, ..., x_N \text{that are } \leq x}{N}, \tag{1}$$

where $x = (x_1, x_2, ..., x_N)$ is a sample of size N, and the null distribution (that is, the distribution from which the null hypothesis claims the sample is), $F(x)$. The test can also be used as a two-sample test; in that case, the test statistic is the maximum distance between the ECDF of the first sample and the ECDF of the second sample.
The KS statistic, K, for both cases is illustrated in Figure 2.1.

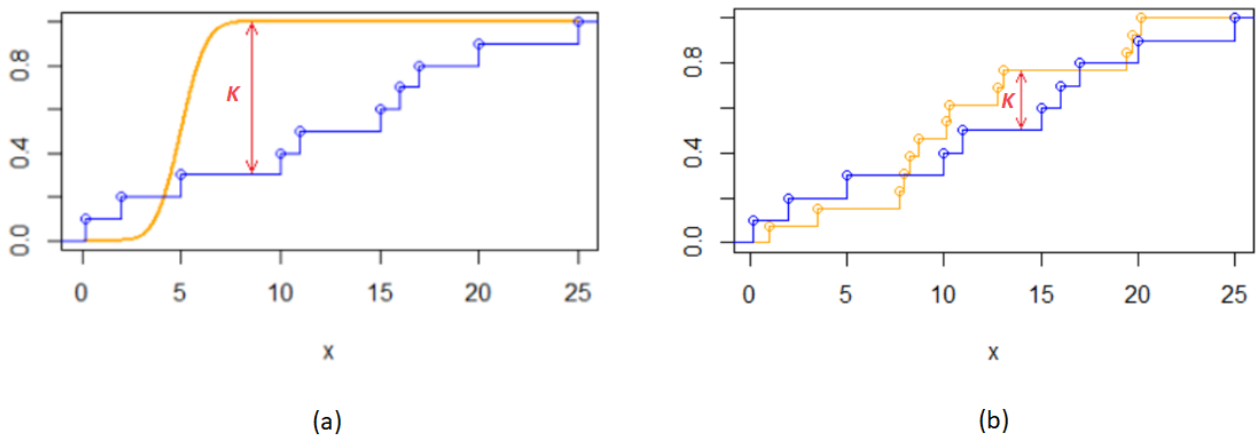

Figure 2.1: One-dimensional KS statistic: (a) 1-sample case, (b) 2-sample case

The KS test is a nonparametric test; no major assumptions are made *a priori* about the sample distribution.

Its main advantages are that it is easy to understand and use.

Power is a common way of evaluating how good a test is. It is the probability of rejecting the null hypothesis when it is false, that is, it is $1-\beta$, where β is the probability of a type II error. The higher the power is, the better the test is. 80% is sometimes mentioned as a general rule of thumb for what constitutes the lower bound on power for a "good" test. Figure 2.2 shows the power of (one-dimensional) KS tests involving normal distributions. Power for sample sizes N of 1000, 10,000, and 100,000 is shown as a function of b, the mean of the normal distribution the sample is from, the null distribution being a standard normal; the level of significance, α, is 0.01.

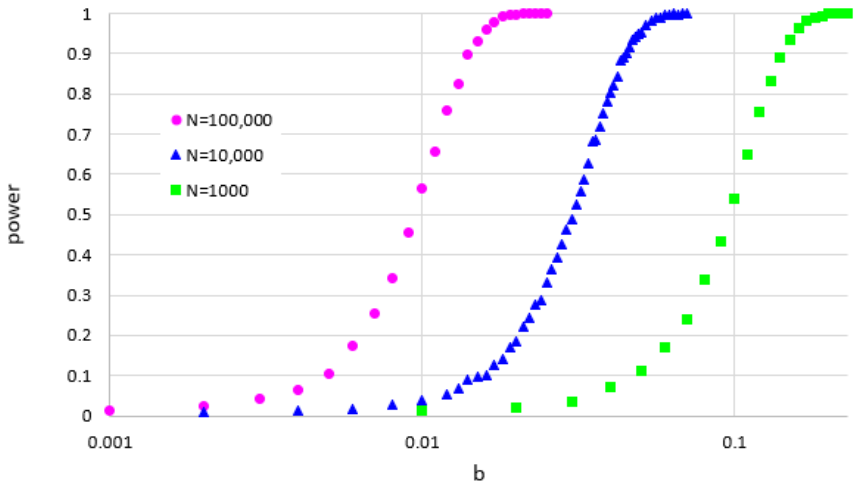

Figure 2.2: Power of the (one-dimensional) KS test as a function of b; $\alpha = 0.01$.

The power of the 1D KS test is less than that of a corresponding Z-test, as illustrated in Figure 2.3. This is not surprising since the Z-test assumes normality, but the KS test makes no such assumptions.

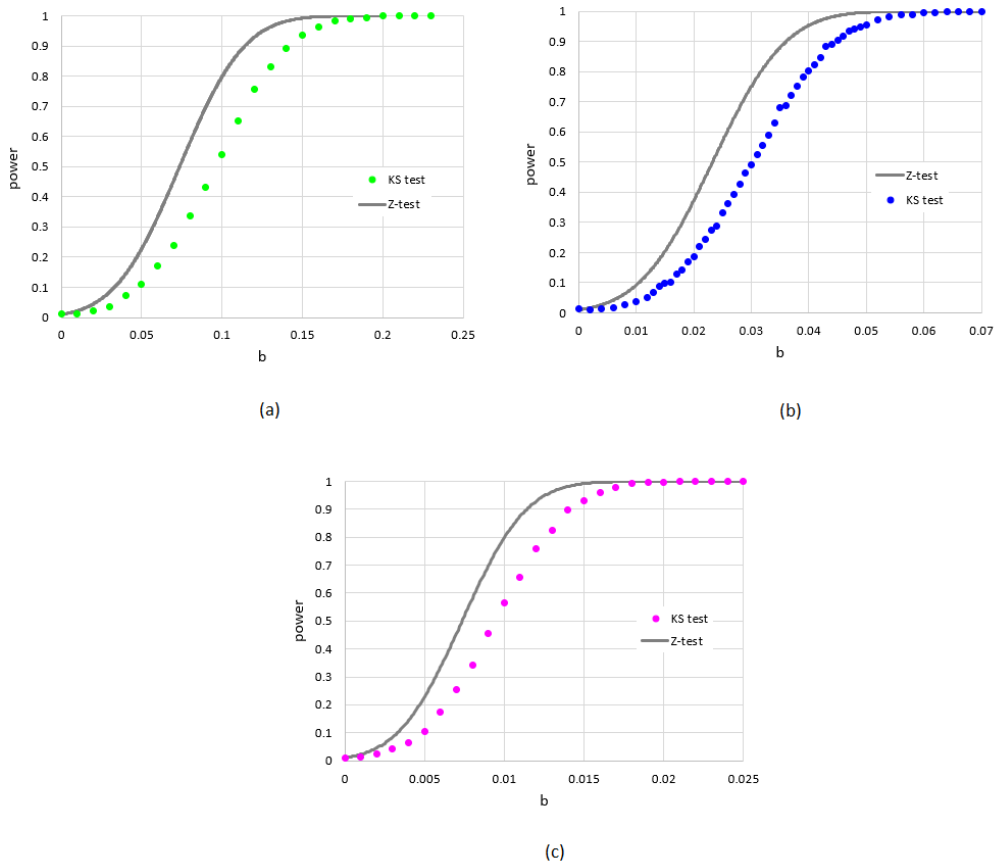

Figure 2.3: Comparison of the power of the (one-dimensional) KS and Z tests as a function of b: (a) for N=1000, (b) for N=10,000, (c) for N=100,000; $\alpha = 0.01$

The cumulative distribution function of the Kolmogorov statistic, K, is given by (e.g., Marsaglia *et al.*, 2003)

$$\lim_{N\longrightarrow\infty} \Pr(\sqrt{N}K \leq x) = 1 - 2\sum_{i=1}^{\infty}(-1)^{i-1}e^{-2i^2x^2} \tag{2a}$$

$$= \frac{\sqrt{2\pi}}{x}\sum_{i=1}^{\infty} e^{-(2i-1)^2\pi^2/(8x^2)} \tag{2b}$$

The representation of the Kolmogorov distribution as N goes to infinity was imple-

mented using code by Kapelner (2020). Before the code was used, its output was checked against this author's recursive implementation of Equation (2a) in Excel; the match was excellent, so Kapelner's code was incorporated into the R scripts used for this work (see Appendix).

The probability distribution function (PDF) and cumulative distribution function (CDF) of the Kolmogorov distribution are shown in black in Figure 2.4; the colored curves in part (b) illustrate the CDF of the Kolmogorov distribution raised to some other powers, used in Section 3.2.

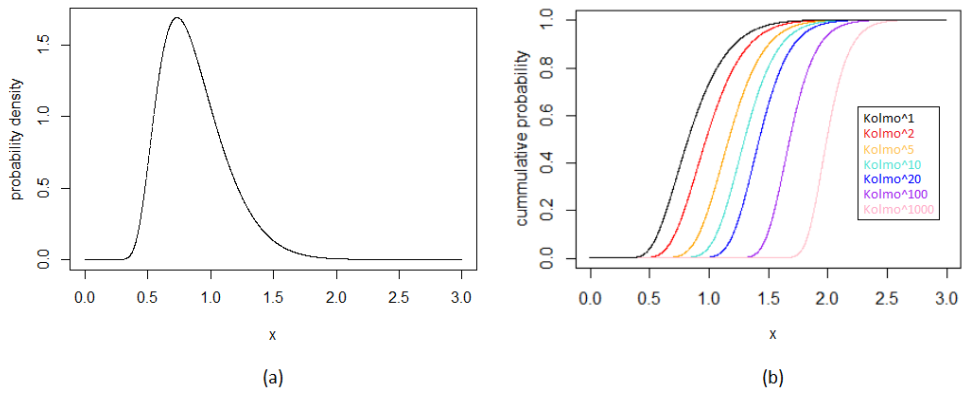

Figure 2.4: PDF of the Kolmogorov distribution; (b) CDF of the Kolmogorov distribution raised to the powers 1, 2, 5, 10, 20, 100, and 1000

The KS test is one of the so-called EDF (empirical distribution function) tests, because it uses an ECDF, as opposed to other goodness-of-fit tests like the χ^2 test. Stephens (1974) reports that, if the distribution is continuous and completely specified, "EDF statistics give more powerful tests of H_o than χ^2. The disadvantage is that they are neither well adapted for discrete distributions, nor for the case when parameters must be estimated from the sample."

Other EDF tests are the Cramér-von Mises test (Cramér, 1928 and von Mises, 1928) and the Anderson-Darling (AD) test (Anderson & Darling, 1952), which are members of the same family. The AD test pays more attention to the tails of the distribution than the KS

test. It is not fully distribution free, since the critical values depend on the distribution being tested (Kvam & Vidakovic, 2007). Not surprisingly, this loss of generality leads to an increase in power compared to the KS test, as reported, for instance, by Boyerinas (2016).

Even though it is usually considered a one-dimensional test, the KS test can be extended to higher dimensions. The multivariate ECDF must be computed for the sample(s), which gets quite involved in higher dimensions (see, for example, Bentley, 1980, or Scott, 2015). Justel *et al.* (1997) propose an algorithm for computing the bivariate Kolmogorov-Smirnov statistic and conclude that numerical complications for application to higher dimensions than 2 seem considerable.
Therefore, a quicker and easier test would be welcome in some situations, which is why we investigated the projective KS test, introduced in the next section.

3 Projective KS Test

3.1 Description

The basic idea of the projective KS test is the projection of the d-dimensional data onto a unit vector in the direction of a random line through the origin, repeated for any desired number L of such random lines (either because the data are partitioned into L sub-samples, each projected once, or because the whole sample is projected onto L lines, as will be explained shortly). The projections can be done in any dimension d. An example for $d = 2$ is shown in Figure 3.1.

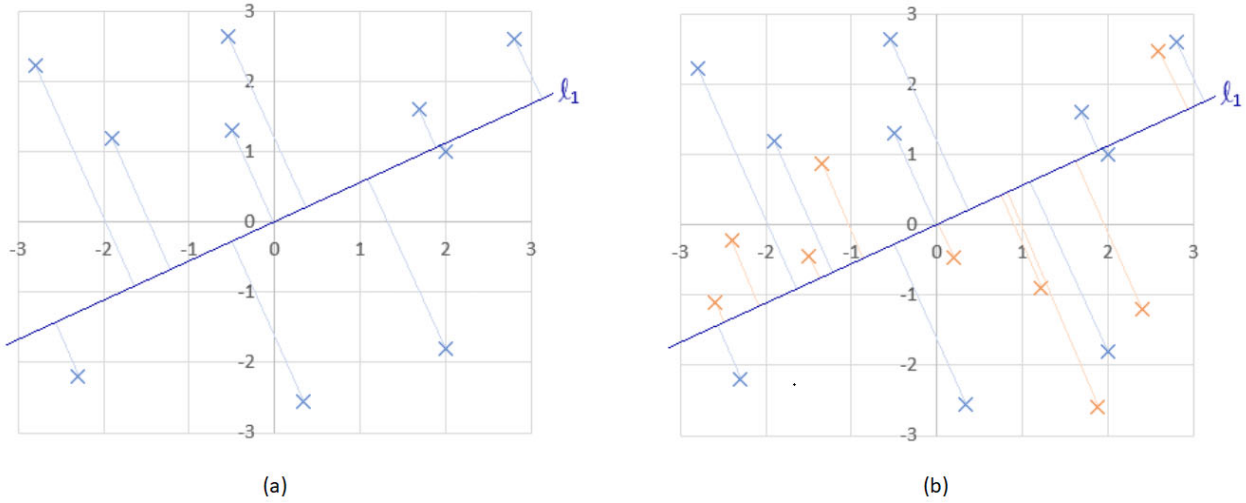

Figure 3.1: Projection of the sample(s) onto a random line ℓ_1: (a) 1-sample case, (b) 2-sample case

No matter what d is, the result of the projections is that the problem has basically been simplified from a challenging d-dimensional KS test to L one-dimensional KS tests yielding L KS statistics: $K_1, K_2, ..., K_L$.

The test statistic for the projective KS test, $\hat{K}$, is defined as the maximum of the L one-dimensional KS statistics obtained from the L projections, scaled by the size of the projected sample or sub-sample.

More detailed descriptions of the projective KS test will be given below for each of two ways of implementing it:

- Method A, which involves randomly partitioning the sample into L sub-samples and projecting each sub-sample onto one line, and
- Method B, which involves projecting the whole sample onto L lines.

The projective KS statistic, $\hat{K}$, only goes to zero if all the K_i go to zero.

Kolmogorov (1933, translation available in Shirayev, ed, 1992) showed that one-dimensional KS statistics go to zero in the limit of $N \longrightarrow \infty$ if the sample comes from the hypothesized distribution, no matter what the distribution is, as long as it is continuous.

This work is based on the validity of the statement that the projective KS statistic is arbitrarily small if and only if X_n, a sample of size n, converges in distribution to X, whose distribution is the hypothesized distribution:

$$\hat{K} \longrightarrow 0 \quad \Leftrightarrow \quad X_n \xrightarrow{dist} X, \tag{3}$$

which is supported by the following proof (more general proofs can be found, for example, in Walther, 1997 or Lyons & Zumbrun, 2018; both are based on Cramér & Wold, 1936):

$\Rightarrow$

Assume the projective KS statistic is zero:

$$\hat{K} = 0. \tag{4}$$

That means

$$\max(K_1, K_2, ..., K_L) = 0. \tag{5}$$

(even as $L \longrightarrow \infty$). Since the maximum is zero, K (K_i for any i) is zero for all possible unit vectors $\hat{u}$, (that is, for all possible directions), so the one-dimensional KS test statistic goes to zero for the projection of X_n and X onto any line, that is,

$$K = \max \left\| F_{X_n \cdot \hat{u}} - F_{X \cdot \hat{u}} \right\| = 0, \tag{6}$$

for all $\hat{u}$ such that $|\hat{u}| = 1$. ($F_{X_n \cdot \hat{u}}$ denotes the ECDF of the projections of X_n onto $\hat{u}$ and

$F_{X \cdot \hat{u}}$ denotes the CDF of the projections of X onto $\hat{u}$.)

So the distribution of the projections of X_n is the same as the distribution of the projections of X for any line, that is, for any unit vector $\hat{u}$,

$$f_{\hat{u} \cdot X_n}(z) = f_{\hat{u} \cdot X}(z). \tag{7}$$

Next, it will be shown that the distribution of the projections being the same implies that the original ("un-projected") sample comes from the hypothesized ("un-projected") distribution.

Since $f_{\hat{u} \cdot X_n}(z) = f_{\hat{u} \cdot X}(z)$ for all $\hat{u}$ in $\mathbb{R}^d$ and all z in $\mathbb{R}^d$, the characteristic functions of the projections are the same:

$$\phi_{\hat{u} \cdot X_n}(z) = \phi_{\hat{u} \cdot X}(z), \tag{8}$$

so, by definition,

$$E[e^{it\hat{u}X_n}] = E[e^{it\hat{u}X}] \tag{9}$$

for any t.

Let $a = \hat{u}t$. Then

$$E[e^{iaX_n}] = E[e^{iaX}] \tag{10}$$

is true for any a.

We can rename a as t if we want to, since it is arbitrary. Then

$$E[e^{itX_n}] = E[e^{itX}], \tag{11}$$

so the characteristic function of X_n is equal to the characteristic function of X, that is, $\phi(X_n) = \phi(X)$, so the distribution of X_n and the distribution of X are the same. Expressed differently,

$$X_n \xrightarrow{dist} X, \tag{12}$$

that is, the sample, X_n, converges to the random variable X in distribution, where X follows the hypothesized distribution, f_X.

$\Leftarrow$

Conversely, assume X_n converges to X in distribution:

$$X_n \xrightarrow{dist} X, \tag{13}$$

Then

$$F_{X_n} = F_X. \tag{14}$$

Therefore, $F_{X_n} - F_X$ is zero everywhere,
which implies that $F_{X_n \cdot \hat{u}} - F_{X \cdot \hat{u}}$ is zero everywhere,
so that the one-dimensional KS statistic is zero for any unit vector $\hat{u}$ and therefore any line through the origin, so the value of K_i is zero for all i, and therefore the maximum value of K_i, $(i = 1, 2, ..., L)$ is zero, so

$$\hat{K} = 0. \tag{15}$$

■

The work below is bivariate unless otherwise indicated, but the R code (see Appendix)

is set up for any dimension d.

In this work, attention is focused on 1-sample tests rather than two-sample tests, and normal distributions are used: the hypothesized distribution is a d-dimensional standard normal and the sample is drawn form a normal distribution whose mean is a d-dimensional vector whose entries are all zeros except for the first, which is b, and whose variance is the d by d identity matrix.
It would be interesting to consider other cases in a follow-up study.

3.2 Method A: Sub-Sampling

Method A uses random partitioning of the original sample of size N into L subsets ("sub-sampling" below will refer to this random partitioning). This process is equivalent to sampling without replacement and guarantees that the one-dimensional KS statistics obtained, K_i, are independent and identically distributed (i.i.d.), assuming the samples are i.i.d. to begin with.

The procedure for Method A is as follows: for each sub-sample x_i, a random unit vector $\hat{u}_i$ is generated (in the direction of the corresponding random line ℓ_i) and the sub-sample is projected onto it. Then, for each (i^{th}) sub-sample, one calculates K_i, the one-dimensional KS statistic, which is the greatest distance that occurs between the ECDF of the projection of the sub-sample and the projection of the hypothesized (null) distribution:

$$K_i = \max_x \left\| F_{X_n \cdot \hat{u}_i} - F_{X \cdot \hat{u}_i} \right\|. \tag{16}$$

This procedure is followed for each of the L sub-samples (again, one projection per sub-sample). The distribution of these K_i, after scaling by $\sqrt{N/L}$, is the Kolmogorov distribution, as can be seen in the example shown in Figure 3.2. This is true because the

sub-samples are independent.

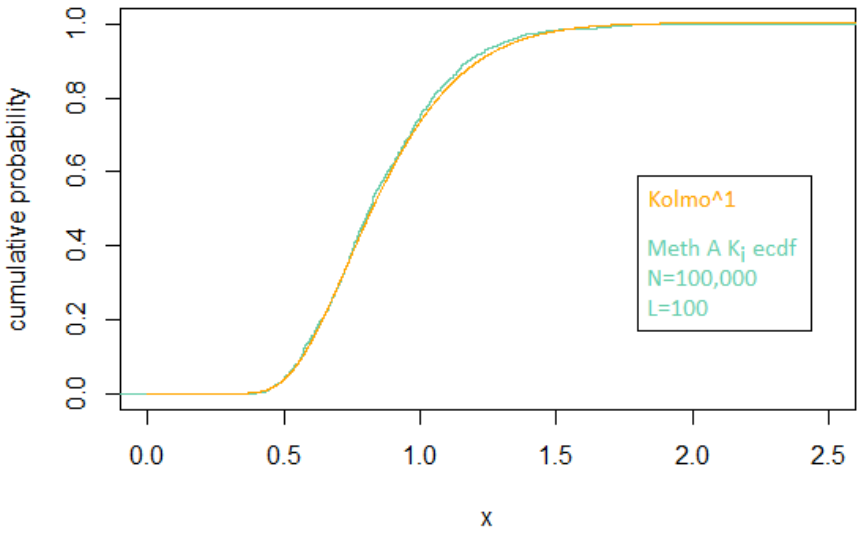

Figure 3.2: ECDF of the scaled K_i obtained for a run with N=100,000 and L=1000 compared to the Kolmogorov distribution

For Method A, we define

$$\hat{K} = max(K_1, K_2, ..., K_L)\sqrt{N/L}. \tag{17}$$

Since the K_i are independent, the distribution of their maximum is the product of the distributions (this is an established fact, but is shown here for completeness):

Let $V_1, V_2, ..., V_n$ be independent random variables.

We want to establish what the PDF of $\max(V_1, V_2, ..., V_n)$ is.

Let $Z = \max(V_1, V_2, ..., V_n)$.

Then if $Z \leq z$ for some z, then $V_i \leq z$ for all i.

In terms of probability,

$P(Z \leq z) = P(V_i \leq z)$ for all i.

But $V_1, V_2, ..., V_n$ are independent, so

$$P(Z \leq z) = \prod_{i=1}^{n} P(V_i \leq z).$$

By definition of the CDF, $P(V_i \leq z) = F_x(z)$, so we get the CDF of $\max(V_1, V_2, ..., V_n)$:

$$F_z(z) = P(Z \leq z) = \prod_{i=1}^{n} F_i(z) \tag{18}$$

In this case, since $K_i\sqrt{N/L}$ follow the Kolmogorov distribution, the maximum of all the scaled K_i will be distributed like the product of the Kolmogorov distribution, that is, using Equation (2a), in the limit as N goes to infinity:

$$\Pr(\hat{K} \leq x) = \left(1 - 2\sum_{i=1}^{\infty}(-1)^{i-1}e^{-2i^2x^2}\right)^L \tag{19}$$

or, informally, $\hat{K} \sim (Kolmo)^L$, where Kolmo denotes the Kolmogorov distribution. This is illustrated in Figure 3.3 for the 2-, 10-, and 100-dimensional cases.

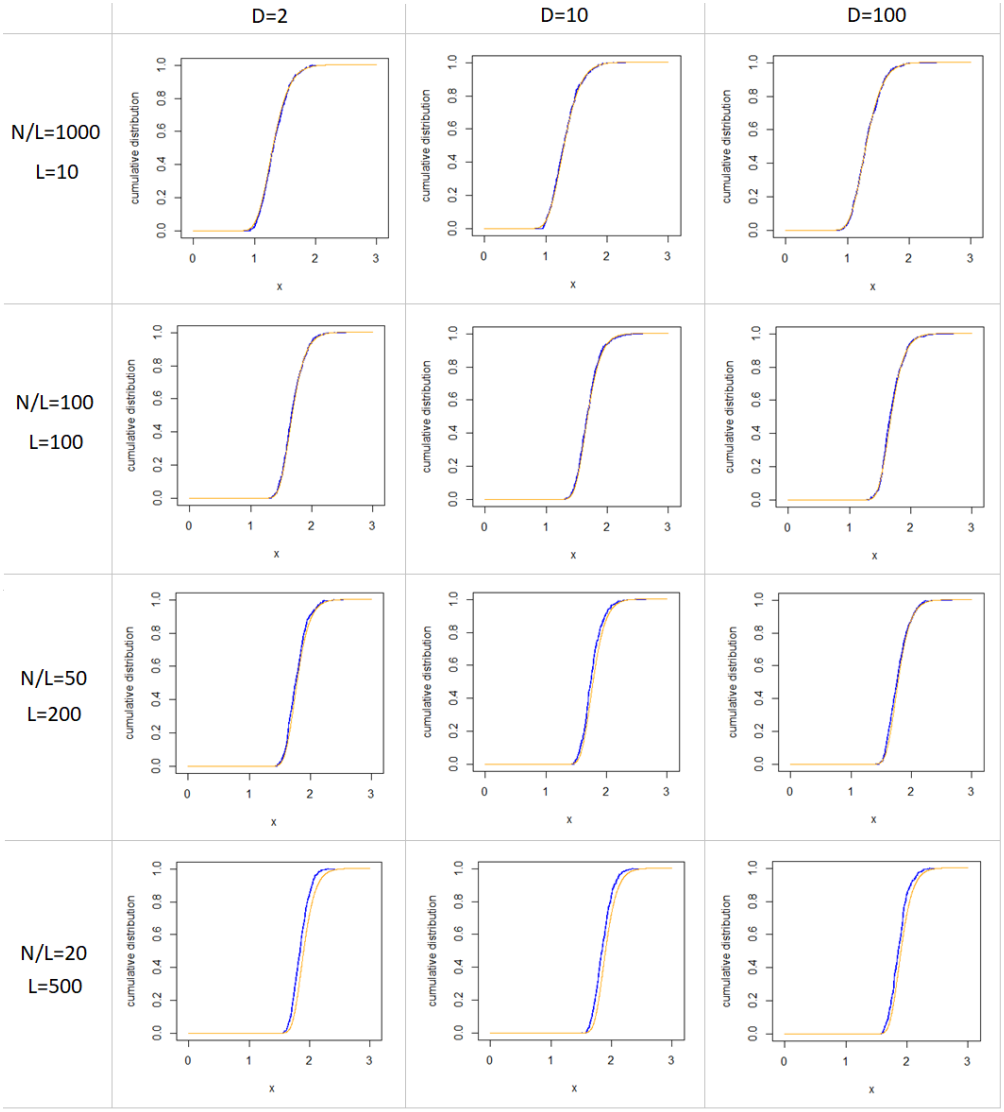

Figure 3.3: ECDF of $\hat{K}$ compared to the Kolmogorov distribution for 3 different N/L; N fixed at 10,000; d=2, d=10, and d=100

The agreement starts deteriorating below an N/L ratio of 100; therefore, sub-sample sizes much smaller than 100 should be avoided.

To evaluate how "good" the test is, power was obtained for different values of L. In order to obtain power, the critical value for a desired maximum probability of type I error must first be obtained. The critical values for different numbers of sub-samples are shown

in Table 3.1 for a level of significance of 0.01. In other words, the area under the curve of the L^{th} power of the Kolmogorov distribution to the right of the critical value is 0.01.

Table 3.1: Critical values (crit_x in the code) for different number of sub-samples (L) for a level of significance (α) of 0.01, Method A

L	critical value
1	1.628
2	1.730
5	1.858
10	1.949
40	2.119
100	2.225

Power as a function of the distance b was then obtained by computing the area under the "empirical PDF" of the projective KS statistics for a particular value of b to the right of the critical value, that is, one minus the value of the ECDF at the critical value.
As expected, the test is more powerful for larger sub-samples, that is, for larger values of N/L, as can be seen in Figure 3.4.

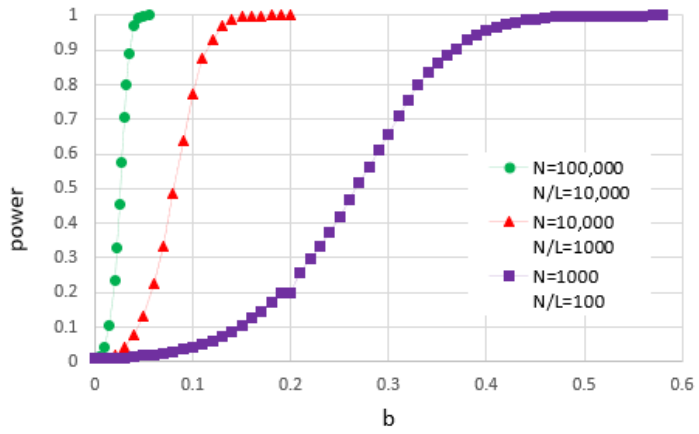

Figure 3.4: Power for Method A as a function of b for different sample sizes; L fixed at 10, α=0.01 (each data point is the average of 4 power run results, each involving 400 $\hat{K}$)

On the other hand, the behavior of the power as a function of b (the difference between the means), is more complicated when N is kept constant and L varies (Figure 3.5), at least with the normal distributions considered in this work. The power rises more sharply with b as L gets smaller; however, for small numbers of sub-samples, the power curve flattens early. This effect is quite significant for $L = 1$ (one sub-sample made up of the whole sample), which sees its power curve rise sharply, but flatten quite a bit when it reaches the vicinity of 0.8, and progress towards 1 very slowly after that.

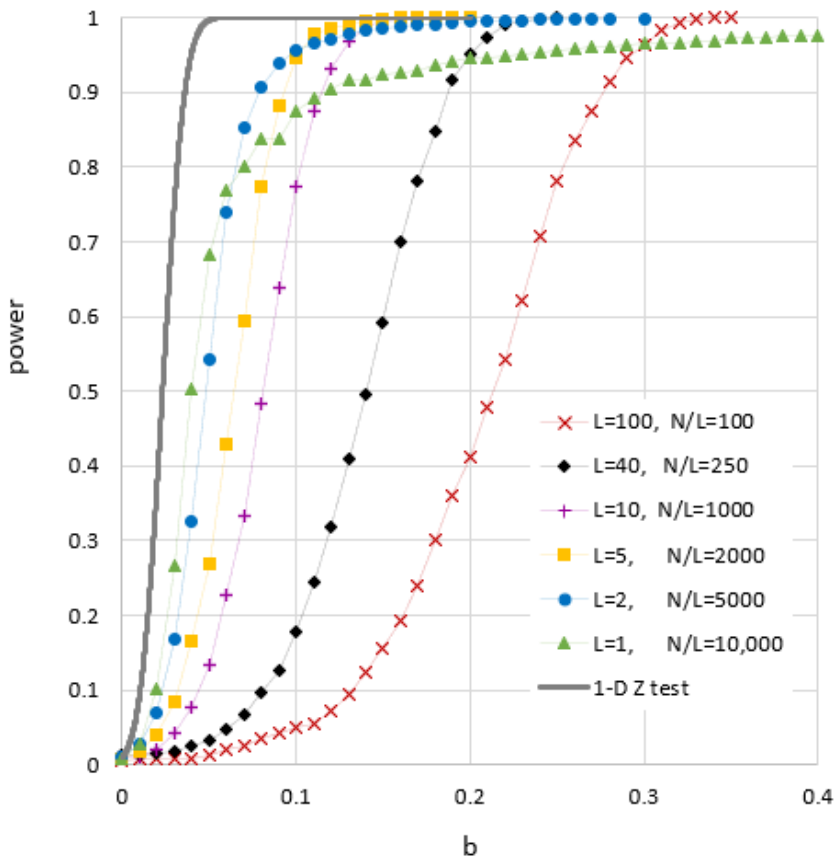

Figure 3.5: Power for Method A as a function of b for different sub-sample sizes; N fixed at 10,000, α=0.01 (each data point is the average of 4 power run results, each involving 400 $\hat{K}$)

So, for the cases studied, results of Method A exhibit an undesired feature: power

does not increase monotonically with L.

When N/L, the sub-sample size, is kept constant at 100, the maximum slope of the power vs. b curve is less steep for smaller L, as shown in Figure 3.6. As L decreases, the power approaches 1 more and more slowly with increasing b.

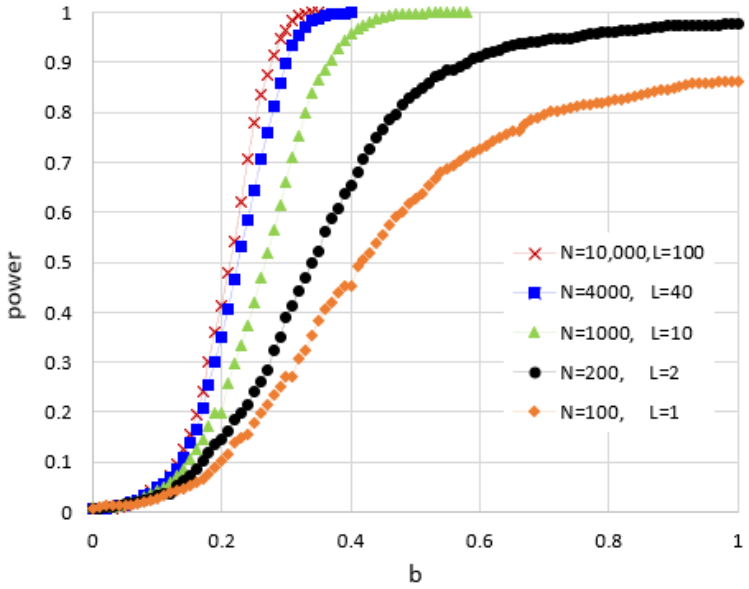

Figure 3.6: Power for Method A as a function of b for different numbers of sub-samples; N/L fixed at 100, α=0.01 (each data point is the average of 4 power run results, each involving 400 $\hat{K}$)

COMPARISON WITH Z-TESTS

Since we are dealing with two normal distributions, the power of a corresponding Z-test was calculated for comparison purposes. The Z-test is one-dimensional, so it can be seen as a "slice" of the bivariate distributions of interest, since they are isotropic.

The power for this case is calculated using

$$\text{power} = \Phi(z_\alpha - b\sqrt{N}). \tag{20}$$

The power of the a corresponding Z-test as a function of b for a sample size of 10,000 is shown in Figure 3.5 for comparison purposes.

For further comparison, the power of the projective KS test, Method A, for sample sizes of 1000 and 100,000 with 10 sub-samples in each case is shown along with the power of a corresponding one-dimensional Z-test in Figures 3.7 and 3.8, respectively .

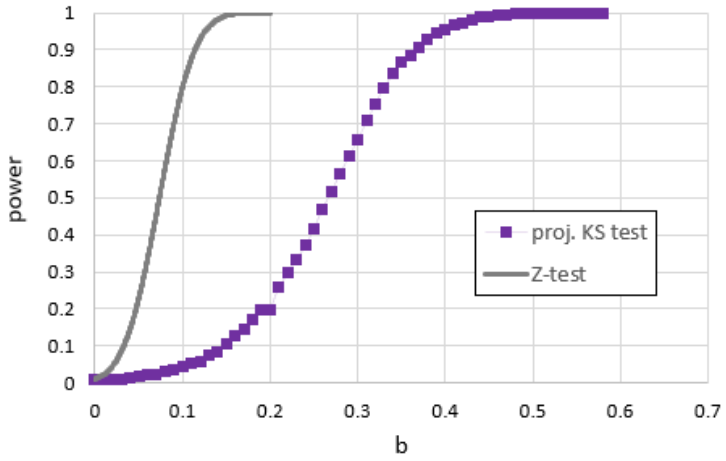

Figure 3.7: Power of the bivariate projective KS test for a sample size of 1,000 and 10 sub-samples compared to the power of the 1D Z-test for the same sample size

As expected, the power goes up with the size of the sample for both methods. The power of the Z-test remains higher, which is explained by the fact that the Z-test has more constraints (in particular, both distributions must be normal).

An interesting observation is that, in these examples, the ratios of the b needed for some power stay close to constant: for example, for $N = 10,000$, a power of 0.8 is reached at $b = 0.0317$ for the Z-test and $b = 0.102$ for Method A of the projective KS test, yielding a ratio of 3.2. The ratios for $N = 1000$ and $N = 100,000$ are 3.25 and 3.2, respectively. For a power of 0.9, the ratios are all 3.2. The ratios across different values of N are consistent; they get somewhat higher for smaller powers (for example, about 3.6 for a power of 0.4).

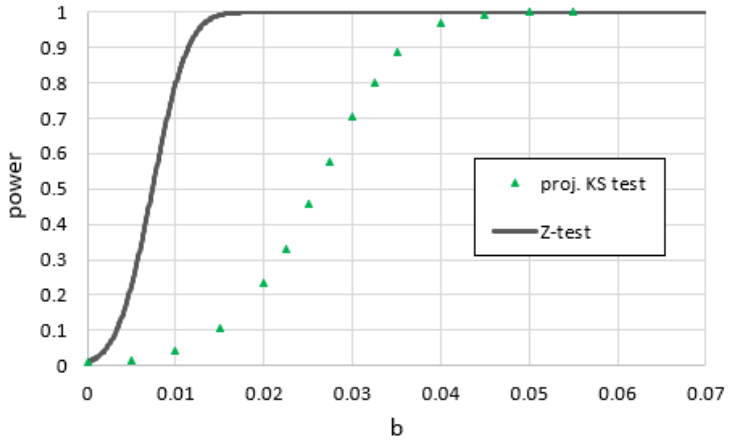

Figure 3.8: Power of the bivariate projective KS test for a sample size of 100,000 and 10 sub-samples compared to the power of the 1D Z-test for the same sample size

3.3 Method B: No Sub-Sampling

The second method that was implemented, Method B, is based on the idea of projecting the whole sample together. One generates a random unit vector and projects the whole sample onto it. Then one calculates K_i, the one-dimensional KS statistic, which is the greatest distance that occurs between the ECDF of the sample and the CDF of the hypothesized (null) distribution. This procedure is repeated for L different randomly chosen unit vectors. These K_i, after scaling by $\sqrt{N}$, will not follow the Kolmogorov distribution, as seen in the example shown in Figure 3.9, because they are not independent, as the same points are used in the calculation of the different K_i.

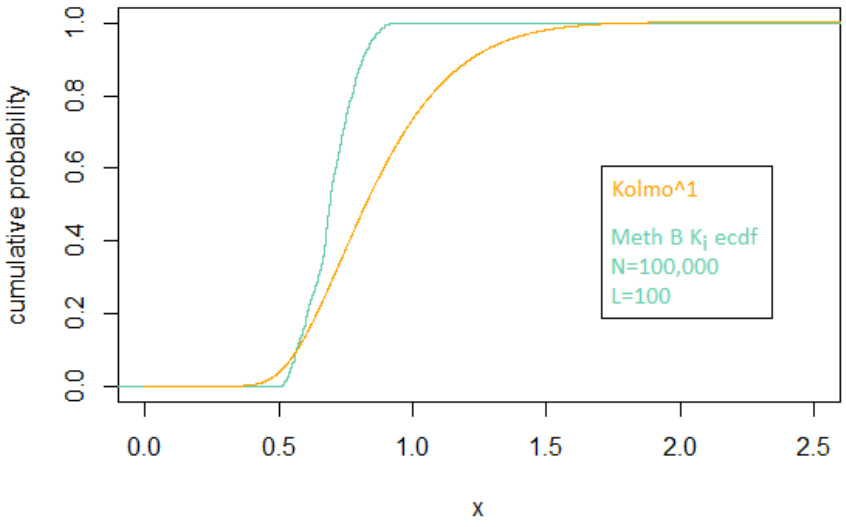

Figure 3.9: ECDF of the scaled K_i obtained for a run with $N = 100,000$ and $L = 1000$ compared to the Kolmogorov distribution

For Method B, we define

$$\hat{K} = max(K_1, K_2, ..., K_L)\sqrt{N}. \tag{21}$$

The major complication in this method is that, unlike in Method A, the distribution of $\hat{K}$ is not known. It is definitely not $Kolmo^L$, as confirmed by the example presented in Figure 3.10, where it can be seen that the ECDF for $L = 4000$ and various values of N lie somewhere between $Kolmo^1$ and $Kolmo^{4000}$. In this example, it is also interesting to note the variation in the ECDF.

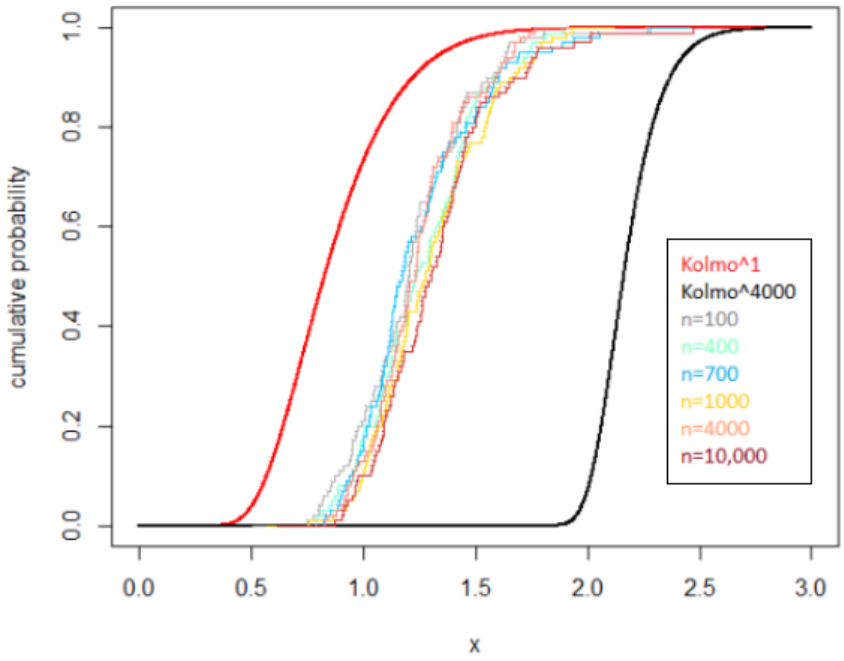

Figure 3.10: ECDF of the $\hat{K}$ obtained for L=4000 and N=100, 400, 700, 1000, 4000, and 10,000 compared to the CDF of the Kolmogorov distribution raised to the powers 1 and 4000

Consequently, the Monte-Carlo method was used to obtain the distribution of the $\hat{K}$, as well as to compute the power of the tests. The computational cost is therefore much higher than for Method A.

For a level of significance of 0.01, the critical values, that is, the values of x to the left of which 0.99 of the area under the PDF obtained by the Monte-Carlo method lies, are shown in Table 3.2.

Table 3.2: Critical values (crit_x in the code) for different numbers of projection lines (L) for a level of significance (α) of 0.01, Method B

L	critical value
1	1.617
2	1.709
5	1.795
10	1.866
40	1.943
100	1.972

Power as a function of the distance b was then obtained by computing the area under the "empirical PDF" of the KS statistics for a particular value of b to the right of the critical value, that is, one minus the value of the ECDF at the critical value.

Just as was done with Method A, attention was focused on 1-sample tests rather than two-sample tests, and normal distributions were used: the hypothesized distribution is a d-dimensional standard normal and the sample is drawn form a normal distribution whose mean is a d-dimensional vector whose entries are all zeros except for the first, which is b, and whose variance is the d by d identity matrix.

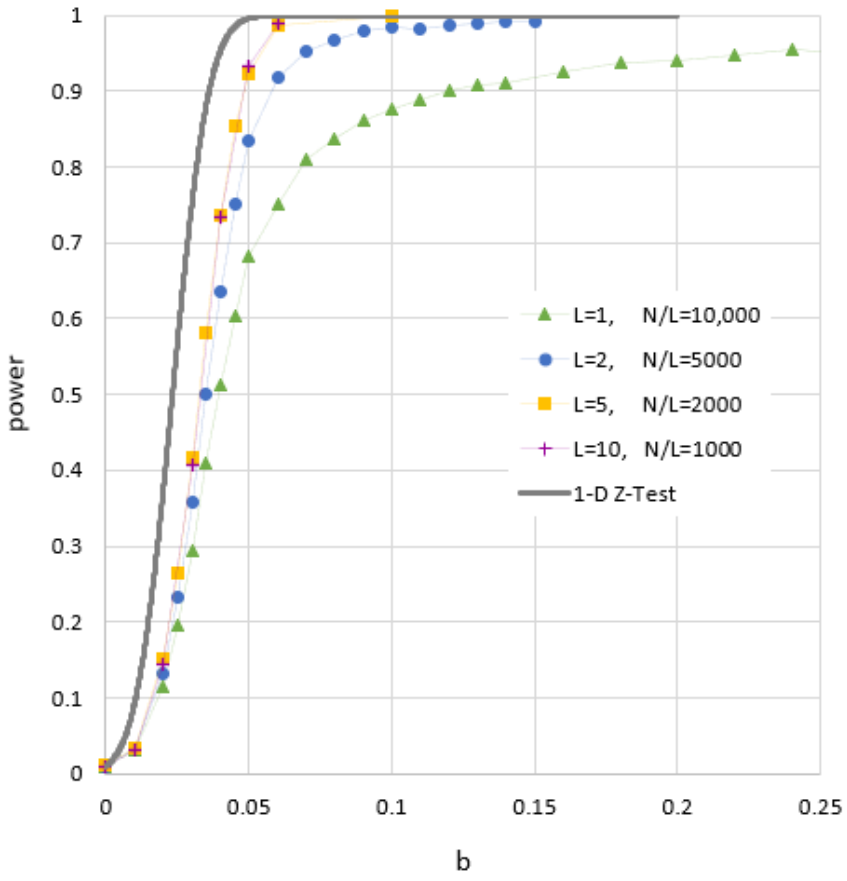

Figure 3.11: Power for Method B as a function of b for different sub-sample sizes; N fixed at 10,000, $\alpha = 0.01$ (50,000 $\hat{K}$ were used for the generation of each critical value; each data point is the average of 4 power run results, each involving 4,000 $\hat{K}$)

For those normals, the power is seen to increase monotonically with b (Figure 3.11). The initial rise in power is about the same for all L (albeit a bit slower for $L = 1$ and $L = 2$), but the power curve flattens early for $L = 1$; for $L = 2$, the curve is closer to higher L values than to $L = 1$. It seems like increasing L beyond 5 does not cause much change. Considering that last trend, the decision was made to stop the numerical experiment, considering how slow the runs for $L = 40$ and $L = 100$ would be.

3.4 Comparisons

PERFORMANCE

The power of Methods A and B was compared for the same conditions, N=10,000 and α=0.01; the results are shown in Figure 3.12.

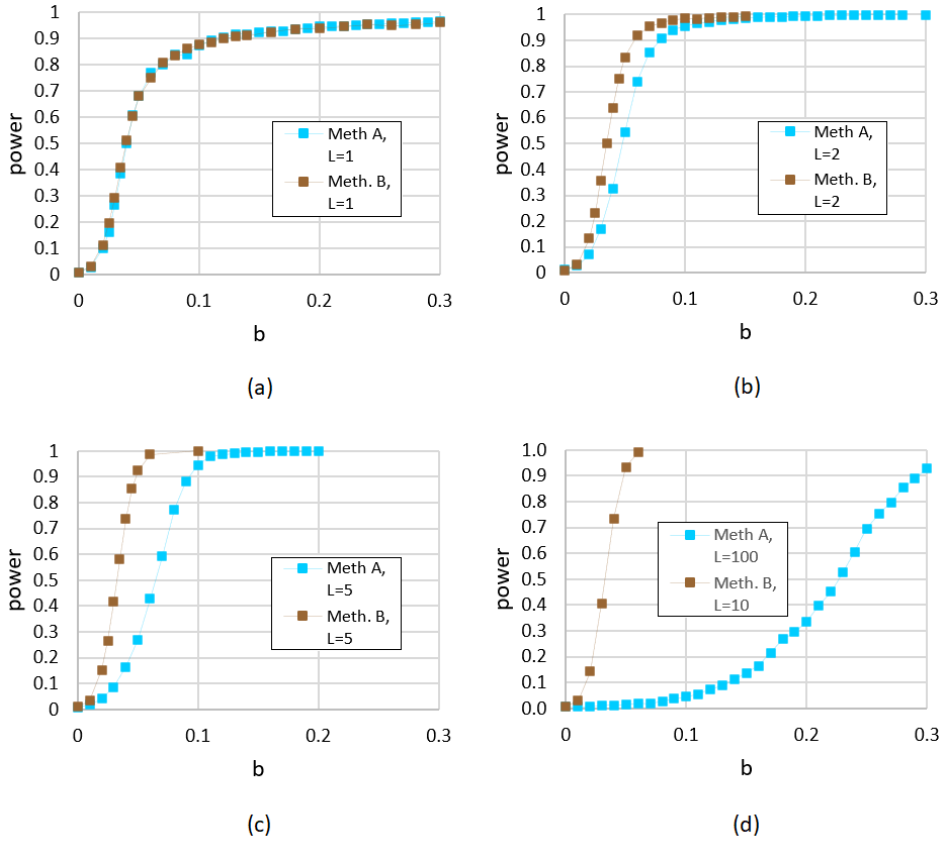

Figure 3.12: Power comparison between Methods A and B: α=0.01, N fixed at 10,000, (a) L=1, (b) L=2, (c) L=5, (d) L=10

Part (a) of Figure 3.12 confirms that, for L=1, Method A and Method B reduce to the same situation. L=1 means no sub-sampling in Method A, so $\sqrt{N/L}K_i = \hat{K}$; on the other hand, L=1 means one projection line in Method B, so one K_i; therefore, $\sqrt{N}K_i$=$\hat{K}$ as well.

As L rises, the power for Method A falls further and further below that of Method B. This is not surprising, since it is well established that power rises with increasing sample size. As L rises, the sub-sample size N/L (the effective sample size for Method A) becomes a smaller and smaller fraction of the overall sample size N (the effective sample size for Method B).
It was also observed that the power increases monotonically with b for Method B, but not for Method A, at least for the normal distributions considered.

COMPUTATIONAL COST

A major difference between Methods A and B is the computational cost involved in applying them. When running the experiments, it was very clear that Method B ran significantly more slowly than Method A for a comparable problem.
In very broad terms, the most time-consuming action in both methods is the sorting involved in running a KS test, which if of order $NlogN$, or O($NlogN$). Method A is O($L\frac{N}{L}log\frac{N}{L}$), that is, O($Nlog\frac{N}{L}$), while Method B is O($LNlogN$). Each would be faster with parallel computing.

4 Discussion

Two versions of the projective KS test were proposed. The first is Method A, which entails randomly partitioning the sample into L equal-sized sub-samples and projecting each sub-sample onto one line. The second version of the projective KS test, Method B, involves projecting the whole sample (no sub-sampling) on L randomly chosen lines. In both cases, a one-dimensional Kolmogorov-Smirnov test is then performed on the projected data, producing L one-dimensional KS statistics $(K_1, K_2, ..., K_L)$.
For Method A, the distribution of the $\hat{K}$ is known, but for Method B, it is not, as the K_i are not independent. Method B was found to have higher power, but to be more expensive, computationally speaking, than Method A.
For a sample size representative of potentially applicable experiments (N=10,000), it was found

that using Method B with an L of 3 or 4 may be the best possible choice for the normal distributions considered here.

There could be cases involving very large sample sizes or high dimensions for which Method A could be considered in order to keep computing time manageable. In those cases, further investigation would be needed to see if a value of L of 2 or 3 would still be recommended for sample sizes larger than 10,000, even for the normal distributions used.

There are several directions in which further work based on the results herein could be pursued.

1. This analysis focused on the one-sample test, but it could be extended to the 2-sample test. It seems like the transition would be seamless for Method A, but not as straightforward for Method B. Similarly, the work could be extended to other distributions than normals.

2. The distribution of $\hat{K}$ for Method B could also be investigated.

3. This work focused on equality of distributions, but the concepts could also be implemented to test the independence of random variables: given joint samples of random variables, (X^1, Y^1), $(X^2, Y^2), ..., (X^n, Y^n)$, where (X^i, Y^i) is the i^{th} sample from the joint distribution of X and Y, they could be used to determine whether X and Y are independent, that is, whether $F_{XY}(x, y) = F_X(x)F_Y(y)$.

4. Computational cost could also be investigated further. It seems certain that a multidimensional KS test would involve a higher computational cost than the projective KS test (either method).

5. An exciting area of future work is the application of the projective KS test to real-life test data, and in particular to neurological data.

The projective KS test could potentially be used as a relatively inexpensive tool (from a computational point of view) to test for equality of distributions and/or random variable independence.

Appendix–R Code

Code for Method A

```
#############################################################################
#
#              Code for Method A, d-D
#
#              Multivariate Normals
#
#
#
#for debugging:
#set.seed(1)
#
#
#
#-------------------------------------------
#  Set variables to desired values
#-------------------------------------------
#
# dimensionality
d<-2
# desired level of significance
alpha<-0.01
# desired number of projective KS statistics (K-hat's) to be generated
num_k_hat<-10000
# sample size
n<-10000
N<-n
# desired number of projections (i.e., of lines (or random vectors)
num_sub<-1
L<-num_sub
# desired non-zero (first) term in the mean of the multivariate
#   normal distribution from which the sample comes
#     (this is for the 1-sample test):
#   in this (early) version, the goal is to check whether that
#   sample comes from the standard multivariate normal
#
b<-0.05
#  x such that the probability in the right tail is alpha
crit_x<-1.628       # crit val for L=1
#
# desired number of power values to be averaged
#  (set to 1 if 1 power value is enough or
#            if power is not of interest;
#    for the thesis, power values were obtained with pow_cnt=4
#    (so they were averages of 4 values))
#    #(the whole script runs this number of times, so do not set
```

```
#    # to zero even if power is not desired)
pow_rdngs<-1
#
#-----------------------------------------
#
#
#
#-------------------------------------------------------------------
#
# The function KS_1D_1S() takes one vector (in this case the vector
#     of projections for a given random vector) and returns the
#     1D KS statistic
###
##
# fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
#
KS_1D_1S <- function(xproj1) {
   #    # order the sample vector, and remove 0 if a vector already
   #    #    starts with 0
   x1<-sort(xproj1)
   if (x1[1]==0) {
      x1<-x1[-1]
   }
   x1
   n<-length(x1);n
   #    # create the y values of the ecdf
   y1<-seq(1,n)*1/n;y1
   #    # create the vector of y values plus 0
   y1_R<-y1
   #   y1_R
   y1aug<-c(0,y1); y1aug
   y1_L<-y1aug[1:(length(y1aug)-1)]
   #   y1_L
   #    # create the vector of known values from the projected
   #    #   given function
   y2<-y1  # they are the same since we are using the same
   #  x points (and therefore the same number
   #  of xpoints) for the 1-sample test
   #
   #*************
   #*************    (Here, the hypothesized distr is the
   #*************     std normal, and a projection of the
   #*************     std normal is the std normal)
   #*************
   F2_orig<-pnorm(x1,0,1)
   #
   # the step below will be more involved with more
   #    complicated distributions than the standard normal
   F2_proj<-1*F2_orig
   #
   #  Calculate the KS statistic (1-D, for the particular vector)
   #
   # First, the vector of differences with the values of the ecdf
   #  just to the left of the actual value of the function:
```

```
   rawdiff_L<-y1_L-F2_proj;rawdiff_L
   # Then, the vector of difference with the values of the ecdf
   #  just to the right of the actual value of the function:
   rawdiff_R<-y1_R-F2_proj;rawdiff_R
   #    # Take the absolute value of the vector of differences
   alldiff<-abs(c(rawdiff_L,rawdiff_R))
   #alldiff
   #    # the 1D KS statistic for the i^th vector, which we
   #    #    call K_i, (it is sometimes called the D statistic,
   #    #    and the name of the variable (set early on) reflects
   #    #    that) is the maximum value in the difference vector
   Dstat<-max(alldiff)
   return(c(Dstat=Dstat,n=n))
}
# fffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
#
#
# Outer loop over pow_cnt: to get pow_rdngs values
#     of the power to average
powpow<-rep(0,pow_rdngs)
#
#
for(pow_cnt in 1:pow_rdngs) {
   #-----------------------------------------------------------------
   # middle loop over the K-hat's: each pass through the loop,
   #   a new sample is generated; it is partitioned into
   #   L sub-samples.   For each sub-sample, a line is generated
   #   and the KS statistic for the sub-sample projected onto
   #   that line is generated.
   #   That gives us L 1D KS statistics (one per sub-sample).
   #   The maximum of these L 1D KS stats is K-hat, the
   #   projective KS statistic for that sample.
   #
   # Initialize the variable that will hold all the K-hat's
   #
   proj_KS_stats<-rep(0,num_k_hat)
   #
   for(ct in 1:num_k_hat) {
      #
      #-------------------------------------------
      #  Generate a sample vector from the multivariate normal
      #      with mean (b,0,...,0) (d-1 zeros) and variance I
      #              (the d-dimensional identity matrix)
      #-------------------------------------------
      #
      mu1<-c(b,rep(0,d-1))
      sig1<-diag(d)
      #
      #
      # we will use mvrnorm in the MASS package
      library(MASS)
      #
      # Generate the sample
      xraw1<-mvrnorm(n,mu1,sig1)
```

```
      #
      #-----------------------------------------
      #  Inner loop: calculating the 1D KS stats
      #
      #  Initiate the variable that will hold all the
      #      1D KS statistics
      #
      KS_1D_proj<-rep(0,num_sub)
      #
      # loop over number of sub-samples (1 proj per sub-sample)
      #
      for(j in 1:num_sub) {
         # #
         # #
         #
         #
         #   generate the unit vector for this sub-sample
         #     (unit vector in the direction of the line
         #      onto which the sample will be projected)
         #     since, to insure independence, only *one*
         #     projection will be used for each sub-sample
         #     (the angles could also be deterministic)
         #
         proj_vect<-mvrnorm(1,c(1,rep(0,d-1)),diag(d))
         # now normalize
         w<-proj_vect/(sqrt(sum(proj_vect%*%proj_vect)))
         #
         # determine what the jth sub-sample is
         #    (it is the jth "slice" of x1raw)
         subxraw1<-xraw1[((j-1)*n/L+1):(j*n/L),]
         #
         #  # compute the projection of the first sample matrix
         xproj1<-subxraw1%*%w
         #  #
         #
         # we invoke the function KS_1D_1S to get
         #     the 1D KS stat for each sub-sample:
         KS_1D_proj[j]<-KS_1D_1S(xproj1)[1]
      }
      ## end of loop over sub-samples (counter is j)
      #cat('1D KS stats: ',KS_1D_proj,'\n')
      #
      ##  By definition, projective KS statistic for the sample is
      #     the value of the greatest 1D KS statistic out of the L
      #     (or num_sub) 1D KS statistics, one for each sub-sample)
      #
      proj_KS_stats[ct]<-max(abs(KS_1D_proj))
      #
}
   #  End of loop over K-hat's (counter ct)
   cat('Projective KS stat (K-hat): ',proj_KS_stats,'\n')
   #
   ###################################
   #
```

```
#   # ## now, the code to get the pdf and cdf of the Kolmogorov
#   #  ##    distribution,
#   written by Adam Kapelner (see reference in thesis):
#
res = 1e-3
xmax = 3
jmax = 1e4
js = 1 : jmax
xs = seq(from = 0, to = xmax, by = res)
Fx = array(NA, length(xs))
sqrt_two_pi = sqrt(2 * pi)
for (i in 1 : length(xs)){
   x = xs[i]
   Fx[i] = sqrt_two_pi / x * sum(exp(-(2 * js - 1)^2 * pi^2 / (8 * x^2)))
   if (i %% 100 == 0){ cat("F(", x, ") =", Fx[i], "\n") }
}
pacman::p_load(ggplot2)
fx = array(NA, length(xs))
for (i in 1 : (length(xs) - 1)){
   fx[i] = (Fx[i + 1] - Fx[i]) / res
}
#ggplot(data.frame(xs = xs, fx = fx, Fx = Fx)) + geom_line(aes(x = xs, y = fx)) + xlab("k") + ylab(
#########################
#  Now back to my code
#
# Put the cdf of the Kolmo distr. in the variable Kolmo_cdf_fr_wiki
#   x-axis values are in the variable xs
#
Kolmo_cdf_fr_wiki<-Fx
#
#  Since the K_i (the values in KS_1D_proj) are independent
#     (Method A only!),
#     the cdf of the max of the 1D KS stats is
#     the cdf of the Kolmogorov distribution to the power L
#     (or num_sub--the number of projection lines,
#      that is, the number of sub-samples)
#
cdf_max_indep_1D_KS_stats<-Kolmo_cdf_fr_wiki^num_sub
#
#  Start the plot of the ECDF of the K-hat and the
#     Kolmogorov distr to the power L:
#
plot(xs,cdf_max_indep_1D_KS_stats, main='Ecdf of K-hat and
     (Kolmogorov distr. cdf)^L',xlab='x',ylab='cumulative distribution',
     pch='.')
#
# Generate the ecdf of proj_KS_stats, the K-hat values
#
xkhat<-sort(proj_KS_stats)
if (xkhat[1]==0) {
  xkhat<-xhat[-1]
}
xkhat
nkhat<-length(xkhat);nkhat
```

```
   #    # create the y values of the ecdf
   ykhat<-seq(1,nkhat)*1/nkhat#;ykhat
   #    # create the vector of y values plus 0
   ykhat_aug<-c(0,ykhat)#; ykhat_aug
   #    # create a stepwise function that is the ecdf of sqrt(n) times
   #        proj_KS_stats, since the K-hat include
   #        that scaling, by definition
   ecdf_scld_k_hat<-stepfun(sqrt(n/num_sub)*xkhat,ykhat_aug)
   # Add the ECDF to the plot of the
   #      cdf of the max of the 1D KS stats
   plot(ecdf_scld_k_hat, add = TRUE, col = "blue",pch='.')
   points(xs,Kolmo_cdf_fr_wiki,
               pch='.',col='orange')
   points(xs,Kolmo_cdf_fr_wiki^L,pch='.',col='darkmagenta')
   #
   #
   # Generate the power of the test:
   #
   powpow[pow_cnt]<-1-ecdf_scld_k_hat(crit_x)
}
cat('ecdf method: ',powpow)
cat('for b = ',b,'the average power is ',(sum(powpow))/pow_rdngs)
#
#############################################################################
```

Code for Method B

```
########################################################################## #
#
#              Code for Method B, d-D
#
#              Multivariate Normals
#
#
#
#for debugging:
#set.seed(1)
#
#-------------------------------------------
#  Set variables to desired values
#-------------------------------------------
#
#
# dimensionality
d<-2
# desired level of significance
alpha<-0.01
# desired number of projective KS statistics (K-hat's) to be generated
num_k_hat<-1000
# desired number of data pairs in the sample
n<-10000
N<-n
# desired number of projections (i.e., of lines)
#    (in the 2D case, this was num_thetas)
num_vect<-1
# desired non-zero (first) term in the mean of the multivariate
#   normal distribution from which the sample comes
#     (this is for the 1-sample test):
#   in this (early) version, the goal is to check whether that
#   sample comes from the standard multivariate normal
#
b<-0.05
#
#  x such that the probability in the right tail is alpha
crit_x<-1.617   # for Method B, num_vect (or L)=1
#
# desired number of power values to be averaged
#  (set to 1 if 1 power value is enough or
#            if power is not of interest;
#    for the thesis, power values were obtained with pow_cnt=4
#    #(the whole script runs this number of times, so do not set
#    # to zero even if power is not desired)
pow_rdngs<-1
#
#-------------------------------------------
#
#
```

```
#
#-----------------------------------------------------------------
#  Here, we will use the function KS-1D_1S
#    to generate the 1D KS statistics
#
# The function KS_1D_1S() takes one vector (in this case the
#    projection of the sample onto a given line (a unit vector
#    in the direction of the given line)) and returns the
#    KS statistic, that is, the maximum difference between the
#    empirical cdf of this vector and the cdf of the
#    hypothesised distribution
#
#
# fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
#
#
KS_1D_1S <- function(xproj1) {
   #    # order the vector of the sample projection, and
   #    #   remove 0 if the ordered vector starts with 0
   x1<-sort(xproj1)
   if (x1[1]==0) {
      x1<-x1[-1]
   }
   x1
   n<-length(x1);n
   #    # create the y values of the ecdf
   y1<-seq(1,n)*1/n;y1
   #    # create the vector of y values plus 0
   y1_R<-y1
   #   y1_R
   y1aug<-c(0,y1); y1aug
   y1_L<-y1aug[1:(length(y1aug)-1)]
   #   y1_L
   #    # create the vector of known values from the projected
   #    #   given function
   #    #  for the standard normal, the projected function
   #    #   is still the standard normal!
   y2<-y1  # they are the same since we are using the same
   #  x points (and therefore the same number
   #  of xpoints) for the 1-sample test
   #
   #*************
   #*************    (Here, the hypothesized distr is the
   #*************     std normal, and a projection of the
   #*************     std normal is the std normal)
   #*************
   F2_orig<-pnorm(x1,0,1)
   #
   # the step below will be more involved with more
   #    complicated distributions than the standard normal
   F2_proj<-1*F2_orig
   #
   #
   #  Calculate the KS statistic (1-D, for the particular vector)
```

```
   #
   # First, the vector of difference with the values of the ecdf
   #  just to the left of the actual value of the function:
   rawdiff_L<-y1_L-F2_proj;rawdiff_L
   # Then, the vector of difference with the values of the ecdf
   #  just to the right of the actual value of the function:
   rawdiff_R<-y1_R-F2_proj;rawdiff_R
   #    # Take the absolute value of the vector of differences
   alldiff<-abs(c(rawdiff_L,rawdiff_R))
   #alldiff
   #    # the 1D KS statistic for the i^th vector, which we
   #    #    call K_i, (it is sometimes called the D statistic,
   #    #    and the name of the variable (set early on) reflects
   #    #    that) is the maximum value in the difference vector
   Dstat<-max(alldiff)
   #cat('function running.','Dstat= ',Dstat)
   return(c(Dstat=Dstat,n=n))
}
# fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff
#
#
#
# Do the whole power calculation pow_rdngs times and
#     average the results at the end to get the actual power
#
powpow<-rep(0,pow_rdngs)
#
#  outer loop is run pow_rdngs times to get an average power
for(pow_cnt in 1:pow_rdngs) {
   #
   #-----------------------------------------------------------------
   #   middle loop over the K-hat's: each pass through the loop,
   #   a new sample is generated;
   #
   #
   # Initialize the variable that will hold all the K-hat's
   #
   proj_KS_stats<-rep(0,num_k_hat)
   #
   for(ct in 1:num_k_hat) {
      #
      #
      #-------------------------------------------
      #  Generate a sample vector from the multivariate normal
      #      with mean (b,0,...,0) (d-1 zeros) and variance I
      #              (the d-dimensional identity matrix)
      #-------------------------------------------
      #
      mu1<-c(b,rep(0,d-1))
      # diag(d) creates the d-dimensional identity matrix
      sig1<-diag(d)
      #
      #
      # we will use mvrnorm in the MASS package
```

```
library(MASS)
#
# Generate the sample
xraw1<-mvrnorm(n,mu1,sig1)
#
#--------------------------------
#
# inner loop over number of number of projection vectors
#
#  Initiate the variable that will hold all the 1D KS statistics
#
KS_1D_proj<-rep(0,num_vect)
#
for(j in 1:num_vect) {
   # #
   #
   #-------------------------------------------------------
   #
   #  Looping over the number of vectors
   #
   #-------------------------------------------------------
   #
   #
   #   generate the unit vector for this sub-sample
   #     (unit vector in the direction of the line
   #      onto which the sample will be projected)
   #
   #     Here, the whole sample is projected onto every
   #      vector, which is why the Ki are NOT independent
   #
   proj_vect<-mvrnorm(1,c(1,rep(0,d-1)),diag(d))
   # now normalize
   w<-proj_vect/(sqrt(sum(proj_vect%*%proj_vect)))
   #
   #
   #cat('projection unit vector = ',w,'\n')
   #
   #  # compute the projection of the sample #
   xproj1<-xraw1%*%w
   #
   #-------------------------------------------------------------------------
   #
   # We invoke the function KS_1D_1S to get
   #    KS_1D_proj, the KS test statistic:
   #
   KS_1D_proj[j]<-KS_1D_1S(xproj1)[1]
}
## end of loop over projections (vectors) (counter is j)
#cat('1D KS stats: ',KS_1D_proj,'\n')
#
##  By definition, projective KS statistic for the sample is
#     the value of the greatest 1D KS statistic out of the L
#     (or num_vect) 1D KS statistics, one for each projection vector)
#
```

```
   proj_KS_stats[ct]<-max(abs(KS_1D_proj))
   #
   #
}
#  End of loop over K-hat's (counter ct)
cat('Projective KS stat (K-hat): ',proj_KS_stats,'\n')
#
###################################
#
# ## Now, the code to get the pdf and cdf of the
# ##   Kolmogorov distribution,
#   written by Adam Kapelner (see reference in thesis):
res = 1e-3
xmax = 3
jmax = 1e4
js = 1 : jmax
xs = seq(from = 0, to = xmax, by = res)
Fx = array(NA, length(xs))
sqrt_two_pi = sqrt(2 * pi)
for (i in 1 : length(xs)){
   x = xs[i]
   Fx[i] = sqrt_two_pi / x * sum(exp(-(2 * js - 1)^2 * pi^2 / (8 * x^2)))
   if (i %% 100 == 0){ cat("F(", x, ") =", Fx[i], "\n") }
}
pacman::p_load(ggplot2)
fx = array(NA, length(xs))
for (i in 1 : (length(xs) - 1)){
   fx[i] = (Fx[i + 1] - Fx[i]) / res
}
#ggplot(data.frame(xs = xs, fx = fx, Fx = Fx)) + geom_line(aes(x = xs, y = fx)) + xlab("k") + ylab(
#########################
#  Now back to my code
#
# Put the cdf of the Kolmo distr. in the variable Kolmo_cdf_fr_wiki
#   x-axis values are in the variable xs
#
Kolmo_cdf_fr_wiki<-Fx
#
#   Create an ECDF of the K-hat:
#
xkhat<-sort(proj_KS_stats)
if (xkhat[1]==0) {
  xkhat<-xhat[-1]
}
xkhat
nkhat<-length(xkhat);nkhat
#    # create the y values of the ecdf
ykhat<-seq(1,nkhat)*1/nkhat#;ykhat
#    # create the vector of y values plus 0
ykhat_aug<-c(0,ykhat)#; ykhat_aug
#    # create a stepwise function that is the ecdf of sqrt(n)
#        times proj_KS_stats, since the K-hat include
#        that scaling, by definition
ecdf_scld_k_hat<-stepfun(sqrt(n)*xkhat,ykhat_aug)
```

```
#
#
############################
#
#
#  create a histogram (not a fan of the built-in histogram):
#
#  K-hat's in bins whose size I choose.
#  enter the number of desired bins below:
num_bins<-10
# loop over the K-hat's and put each one in the desired bin,
#   that is, increase the number in each bin by 1
# Now determine the bin width:
bin_width<-(max(xkhat)-min(xkhat))/num_bins
# assign the bin number to each K-hat:
bin<-(xkhat-min(xkhat))%/%bin_width+1
bin[num_k_hat]<-bin[num_k_hat]-1
#bin
# create a vector (length num_bins) that contains the number
#  of K-hat's in each bin
#
# initialize first
k_hat_in_bin<-rep(0,num_bins)
#
for(ii in 1:num_bins) {
   k_hat_in_bin[ii]<-sum(bin==ii)
}
#k_hat_in_bin
# generate the vector of the mid-points of the bins
mid_bins<-seq(1,num_bins)
mid_bins<-min(xkhat)+bin_width*(mid_bins-1/2)
scld_mid_bins<-sqrt(n)*mid_bins
# plot the pdf (histogram w/ num on y axes divided by the number of bins)
#plot(scld_mid_bins,k_hat_in_bin/num_k_hat,col='blue3',pch=20,
#     xlim=c(sqrt(n)*min(xkhat),sqrt(n)*max(xkhat)))
# plot the cdf
plot(scld_mid_bins,cumsum(k_hat_in_bin/num_k_hat),pch=20,
     col='firebrick2',xlim=c(sqrt(n)*min(xkhat),sqrt(n)*max(xkhat)))
#
########
#
#
#  Power calculations
#
#  We just ran num_k_hat samples; we get the percentage
#         of times the statistic K-hat is above crit_x,
#         this will be the power!!
#
#  >> we count the elements of xkhat (the ordered K-hat's)
#      that are above crit_x and divide by num_k_hat
#       That's the power!
#
#  xkhat is the sorted vector of K-hat's, proj_KS_stats
num_bigger_than_crit_x<-findInterval(crit_x,(sqrt(n)*xkhat))
```

```
   powpow[pow_cnt]<-1-num_bigger_than_crit_x/num_k_hat
   }
# end of loop over pow_cnt
cat('For ',b,'the power is ',(sum(powpow))/pow_rdngs)
#
#
#############################################################################
```

References

Anderson, T. W., & Darling, D. A. (1952). Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. The annals of mathematical statistics, 193-212.

Bentley, J. L. (1980). Multidimensional divide-and-conquer. Communications of the ACM, 23(4), 214-229.

Boyerinas, B. M. (2016). Determining the statistical power of the Kolmogorov-Smirnov and Anderson-Darling goodness-of-fit tests via Monte Carlo simulation. Center of Naval Analyses, Arlington, United States.

Cramér, H. (1928). On the composition of elementary errors: First paper: Mathematical deductions, and second paper: Statistical applications. Scandinavian Actuarial Journal, 1928(1), 13-74 and 141-180.

Cramér, H., & Wold, H. (1936). Some theorems on distribution functions. Journal of the London Mathematical Society, 1(4), 290-294.

Justel, A., Peña, D., & Zamar, R. (1997). A multivariate Kolmogorov-Smirnov test of goodness of fit. Statistics & Probability Letters, 35(3), 251-259.

Kapelner, A. (2020). Kolmogorov Distribution PDF. Wikipedia. https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test#/media /File:KolmogorovDistrPDF.png. Retrieved February 7, 2021.

Knuth, D. E. (1998). Art of computer programming, volume 2: Seminumerical algorithms. 3rd Edition. Addison-Wesley.

Kolmogorov, A. N. (1933). Sulla determinazione empirica delle leggi di probabilita. Giorn. Ist. Ital. Attuari, 4, 1-11.

Kvam, P. H., & Vidakovic, B. (2007). Nonparametric statistics with applications to science and engineering (Vol. 653). John Wiley & Sons.

Lyons, R., & Zumbrun, K. (2018). A calculus proof of the Cramér–Wold theorem. Proceedings of the American Mathematical Society, 146(3), 1331-1334.

Marsaglia, G., Tsang, W. W., & Wang, J. (2003). Evaluating Kolmogorov's distribution. Journal of statistical software, 8(18), 1-4.

Scott, D. W. (2015). Multivariate density estimation: theory, practice, and visualization. John Wiley & Sons.

Shiryayev, A. N. (Ed.). (1992). Selected works of A. N. Kolmogorov: Volume II: probability theory and mathematical statistics (Vol. 26). Springer Science & Business Media.

Smirnov, N. (1948). Table for estimating the goodness of fit of empirical distributions. The annals of mathematical statistics, 19(2), 279-281.

Stephens, M. A. (1974). EDF statistics for goodness of fit and some comparisons. Journal of the American statistical Association, 69(347), 730-737.

von Mises, R. E. (1928). Wahrscheinlichkeit, Statistik und Wahrheit. Julius Springer.

Walther, G. (1997). On a conjecture concerning a theorem of Cramér and Wold. Journal of multivariate analysis, 63(2), 313-319.