

RESEARCH ARTICLE

Neutralise: An open science initiative for neutral comparison of two-sample tests

Leyla Kodalci¹  | Olivier Thas^{1,2,3}

¹I-BioStat, Data Science Institute, Hasselt University, Diepenbeek, Belgium

²Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Gent, Belgium

³National Institute of Applied Statistics Research Australia (NIASRA), University of Wollongong, Wollongong, Australia

Correspondence

Leyla Kodalci, I-BioStat, Data Science Institute, Hasselt University, Agoralaan, Gebouw D, B-3590 Diepenbeek, Belgium.
Email: leyla.kodalci@uhasselt.be

Funding information

Hasselt University Grand Challenge BOF, Grant/Award Number: BOF21GP17



This article has earned an open data badge “**Reproducible Research**” for making publicly available the code necessary to reproduce the reported results. The results reported in this article could fully be reproduced.

Abstract

The two-sample problem is one of the earliest problems in statistics: given two samples, the question is whether or not the observations were sampled from the same distribution. Many statistical tests have been developed for this problem, and many tests have been evaluated in simulation studies, but hardly any study has tried to set up a neutral comparison study. In this paper, we introduce an open science initiative that potentially allows for neutral comparisons of two-sample tests. It is designed as an open-source R package, a repository, and an online R Shiny app. This paper describes the principles, the design of the system and illustrates the use of the system.

KEYWORDS

hypothesis tests, neutral comparison, simulations, two-sample problem

1 | INTRODUCTION

In recent years, there have been several calls for neutral comparisons or benchmarks in statistics, bioinformatics, computational biology, and computational sciences in general (Boulesteix et al., 2013; Weber et al., 2019). The core principles of neutral comparison are that computational methods are evaluated without any bias and that the results of this evaluation become public whatever the outcome of the evaluation. This allows for a fair comparison with competitor methods. Despite the growing awareness in the scientific community, still not many initiatives have been taken. In the machine learning area, community challenges are well-known and can be considered as a form of neutral benchmarking (Goodfellow et al., 2013; Hoffmann et al., 2019; Hyndman, 2020): part of a large data set is made publicly available and the challenge is to build a prediction model for predicting a particular outcome. Everyone can apply their choice of machine

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial](https://creativecommons.org/licenses/by-nc/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2024 The Authors. *Biometrical Journal* published by Wiley-VCH GmbH.

learning methods and submit their predictions. These predictions are compared to a private part of the data and results are reported in terms of, for example, sensitivity and specificity. If the challenge is set up correctly, cheating is not possible, and from this perspective the evaluation is neutral. However, there are still some shortcomings: (1) methods are evaluated based on only a single data set; and (2) the results are often not clearly communicated to the wider community. An important nuance in this comparison is the difference in the aim of a community challenge and a neutral benchmark. While a neutral comparison is focused on a fair evaluation of methods with general conclusions, a community challenge aims at solving a modeling task and not specifically benchmarking methods. Many participants consider it rather a game than a scientific endeavor.

In statistical sciences, methods often aim at estimating target parameters or testing hypotheses. Estimators are evaluated in terms of their bias and precision, and hypothesis tests in terms of type I error rate control and power (or false discovery rate and sensitivity in the context of multiple testing). In the statistical science community, papers that propose new methods typically include two ways of evaluating the new methods. The first is to theoretically derive properties, such as consistency of an estimator or test, asymptotic normality of an estimator, a test statistic, and so forth. Many of these properties are of asymptotic nature, and therefore, as a second approach to method evaluation, a simulation study is set up. Simulation studies allow for evaluating methods for finite sample sizes and under various conditions, including settings that deviate from the assumptions underlying the theoretical development of the statistical method. Moreover, competitor methods can also be included in the simulation study, which allows for a direct comparison of methods. Although this may sound as a very fair procedure, in practice it is vulnerable to bias. The number of (potential) simulation scenarios is typically huge and the scientist has to make choices as to what scenarios to include in the paper. However, the danger exists that the scientist may be tempted, directly or indirectly due to cognitive bias, to mostly report results for simulation scenarios that are favorable for her/his new method. The same holds true for the choice of competitor methods. Finally, from the perspective of reproducible science, it is also important that authors make their simulation code publicly available, including the seeds used for initiating the pseudorandom generators. Some authors do, other do not. And even if they do, the code may be hard to run again many years after the publication.

In this paper, we propose a framework for neutral comparisons of statistical hypothesis tests; we do not aim to present results of a comprehensive neutral comparison study. Our initiative makes use of an open-source R package and a repository on GitHub. It is set up as an open science collaboration, and we hope that many statisticians and perhaps journals will make use of it. The central idea is that everyone can submit a new method and/or a new simulation scenario, and the system evaluates (1) the new method on all previously submitted simulation scenarios, and (2) all previously submitted methods on the new scenarios. The simulation results can be posted on the public GitHub repository. In principle, this framework can be tailored to any class of statistical hypothesis tests, but here we present the method for the evaluation of hypothesis tests for the two-sample problem, which is concerned with the null hypothesis that the distribution of an outcome is the same in two populations. We have chosen for this problem because (1) it is an important problem with plenty of applications; (2) a huge number of statistical two-sample tests have been published; (3) all statisticians are familiar with at least some two-sample tests; and (4) the setting is simple so that we can focus here on the neutral comparison methodology without being distracted by the complexity of the statistical methods. The R package also contains functions for visualizing and summarizing the simulation results. These functions are also implemented in an R Shiny app that is run from an R Shiny server (<https://dsi-uhasselt.shinyapps.io/Neutralise/>).

In Section 2, we define the setting for which this initiative has been developed, the two-sample problem. The basic principles of our neutral comparison approach are outlined in Section 3. The design of the R package and the GitHub pages are presented in Section 4. Although it is not the aim of this paper to present the results of a large neutral comparison study, some preliminary results are shown in Section 5. In the first place, they serve to illustrate the R functions for summarizing the results. We formulate some conclusions and recommendations in Section 6 and invite our peers to contribute to this initiative. The Appendix of this paper contains example input and output files of the Neutralise framework, step-by-step instructions to start working with Neutralise, the protocol for the generation of the annual summary report, and brief descriptions of the generalized Tukey Lambda and g -and- h distributions. Specific parameter values of all included parameter settings are not included in the paper because they are *dynamic* in the sense that users can add other values. However, the parameter values can be easily accessed in the NeutraliseFiles repository as well as in the Shiny app. In addition, an annual summary report on the results that were generated and provided in the NeutraliseFiles repository, is added in [supplementary](#).

2 | TWO-SAMPLE PROBLEM

The two-sample problem can be formulated as follows. Let $F_1(\cdot)$ and $F_2(\cdot)$ denote two distribution functions, and let X_1, \dots, X_{n_1} denote a sample of n_1 i.i.d. observations from F_1 , and Y_1, \dots, Y_{n_2} a sample of n_2 i.i.d. observations from F_2 . The most general null hypothesis of interest is $H_0 : F_1 = F_2$, but it may also be defined in a more narrow sense. With $d(\cdot)$ a functional that maps the space of distribution functions to the real line, null hypotheses can also be of the form $H_0 : d(F_1) = d(F_2)$. Examples of functionals include the mean, variance,.... The alternative hypothesis can also range from very general, $H_1 : F_1 \neq F_2$, to narrow, $H_1 : d(F_1) \neq d(F_2)$. Statistical tests are designed to test a two-sample null hypothesis against an alternative. Tests are supposed to be unbiased of size $\alpha \in (0, 1)$, that is, they control the type I error at the nominal significance level α . Sometimes this property can only be proven in an asymptotic sense. Good tests are also supposed to be consistent against a specific fixed alternative, that is, the power of the test converges to 1 when $\min(n_1, n_2) \rightarrow \infty$. Competitor tests that possess these two properties still need to be compared to one another in terms of their small sample properties (type I error rate and power) under a range of conditions; this is typically done in simulation studies. For some testing problems and statistical tests, it can be proven that a test is the uniformly most powerful (UMP) test, which means that the test has largest power among all tests of size α . For testing simple point hypotheses, the Neyman–Pearson lemma shows that the likelihood ratio test possesses this optimality property. For UMP tests, there is no need for empirical evaluation in simulation studies, at least not under the conditions necessary for this property to hold. Unfortunately, the UMP property does not hold for most of the hypotheses that are of interest for the two-sample problem, and hence simulation studies are necessary for the evaluation of the tests.

3 | BASIC PRINCIPLES

Principles of neutral comparison and benchmark studies have been discussed in detail elsewhere; see, for example, Boulesteix et al. (2013) and Weber et al. (2019) and this discussion will not be repeated here. We have selected guidelines as formulated by Weber et al. (2019). Although these were originally developed for benchmarking methods in computational biology, they also suit well for our purposes. In the remainder of this section, we follow these guidelines by formulating answers to several questions.

Formulation of the scope and purpose: Several two-sample problems exist, depending on the formulation of the null and alternative hypotheses. For example, the two-sample t -test can be seen as a method for testing the null hypothesis of equality of means against the two-sided alternative, but in combination with the assumption of normality and equal variances, the null hypothesis coincides with the stronger hypothesis of equality of distributions. The scope and purpose of our study are formulated as follows.

Scope: Comparison of statistical tests that can be used for testing $H_0 : F_1 = F_2$. The alternative hypothesis involves a location shift (LS) (i.e., the interest is in the power of tests as a function of the LS), but also other parameters of the distribution may be altered (e.g., F_1 and F_2 may have different variances).

It is important to note that tests that can be used for testing the null hypothesis of equal distributions can address different distributional aspects and thus have different alternative hypotheses. For example, while the two-sample t -test tests for equality of means (under the assumption of normality and equal variances), the Wilcoxon–Mann–Whitney (WMW) tests for stochastic equality (under the assumption of symmetry and equal shape). Both tests have different target parameters; the difference in means for the two-sample t -test, and stochastic superiority (the probability of a random observation of one group is bigger than the random observation from the other group) for the WMW test. Comparing the results of methods with different alternative hypotheses should be done with caution.

Purpose: Provide the scientific community (1) with an extensive, unbiased, and neutral evaluation of two-sample tests; and (2) with an open-source environment that can reproduce all results and that can be used for evaluating other two-sample tests under other simulation scenarios as proposed by other researchers.

At this point, we want to make the remark that we actually endorse the best statistical practice guideline that statistical tests should be accompanied by an estimate of the effect size and its standard error or confidence interval. However, we leave this as a possible future extension of the framework.

Research team: The composition of the research team is considered important to minimize the perceived bias. Recall that one of the motivations for performing a neutral comparison study, is that many published simulation studies

have been performed by researchers who want to compare their own method with others. The choices made by these researchers may perhaps result in a too optimistic evaluation of their own method. This must be avoided in neutral studies.

One solution exists in having an independent group of researchers performing the benchmarking. In the ideal world, these are scientists who did not develop two-sample tests themselves and who will not benefit from one particular method outperforming another (i.e., no conflict of interest). If this is not possible, then a form of blinding could be applied. These arguments particularly make sense when the neutral comparison study is closed in time and scope, with a publication of the results as the final goal.

Here, we take another route. We argue that the core team that initiates the neutral study is not very important in the sense that they do not affect or steer the results. This is accomplished by adopting an open-ended and open science collaborative effort approach. The core team only makes well-documented open-source computer code available for the benchmarking. The code assures that all results are reproducible and that the results can be published on a web server. The whole scientific community is free to add two-sample tests and simulation scenarios. However, when adding a new statistical test to the system, the test will be evaluated on all simulation scenarios that have been submitted before. Other researchers are free to improve the computer code. The core team, however, gives itself the tasks to promote the use of the Neutralise R package and to disseminate the results as widely as possible.

Selection of methods and simulation scenarios: The selection of methods and simulation scenarios is completely community-driven. As explained before, everyone is free to add methods and simulation scenarios, as long as it is within the scope of this project. When a method is added, it will be evaluated under all previous simulation scenarios, and when a new simulation scenario is added, it will be used for the evaluation of all previous methods.

Selection of evaluation metrics: In general, the choice of evaluation metrics is important as it may favor certain methods over others. However, in the context of hypothesis testing this choice is quite straightforward: tests will be evaluated in terms of the type I error rate control and the power at a fixed nominal significance level. Although the latter may be subject to discussion, we suggest to limit the study to the conventional levels $\alpha = 0.01$, $\alpha = 0.05$, and $\alpha = 0, 10$.

Interpretation of results and formulation of recommendations: As we have designed the neutral comparison study as a dynamic open-ended environment, our primary objective is not to provide interpretations or recommendations ourselves. We rather focus on making all detailed results publicly available and easily accessible, and although we provide a few simple tools for analyzing the results (also available as an interactive Shiny app), we invite the community to add further interactive tools for exploring, summarizing, and visualizing the results. The collection of *neutralized* results is the end product. Summaries will always depend on particular choices and criteria, but the continuous availability of all results, allows everyone to (re)evaluate the methods. However, as suggested by the Reviewers, an annual report will be released in the NeutraliseFiles repository with a curated summary of the results. Where it is not the aim to make recommendations, but rather to guide users through the results. This will not jeopardize the Neutral initiative, as all the results are continuously publicly available. In addition, this annual report will be drawn up according to a predefined protocol to ensure neutral reporting. See Appendix F for more details.

Publishing of the results: Along the lines of the arguments in the previous paragraph, we have no intention to publish the results of the neutral benchmarking study as a journal paper, but rather to make all detailed results public immediately after their computations. The results are always accompanied by their generating R code, pseudorandom number generator seeds, and R session information, making all results reproducible and open for review. We hope that the GitHub repository will become the permanent, incremental online resource for the comprehensive neutral comparisons results for two-sample tests, rather than having (partial) results published as a static journal paper.

Enable future extensions: Since all computer code is open source, the framework can be extended by others. Such extensions can be initiatives of individual researchers who only want to run the extended code on their local computer. However, we invite them to contact the core team so that we can consider including the extension in our code and make it available to the wider community.

From a broader perspective, we hope to see similar initiatives come to life for the evaluation of other classes of statistics tests, or computational methods in general.

Follow reproducible science best practices: Reproducibility of the results is of utmost importance. We have chosen to work with a well-documented publicly available open-source R package in combination with a GitHub repository and an online R Shiny app for the exploration of the results. The simulation results produced by the R package are saved to a directory that also contains all R session information, as well as the R code and seed of the pseudorandom generator that produced the results.

4 | DESIGN

4.1 | General concept

The overarching philosophy is that everyone must be able (1) to add two-sample tests to the system for evaluation, and (2) to add simulation scenarios under which all previously submitted tests will be evaluated. The R package itself is mainly a set of functions that check the eligibility of the submitted methods in terms of the required input (e.g., sample size and data frame) and output (e.g., test statistic and p -value), and, once the submitted methods pass this check, uses the submitted simulation methods to evaluate all tests, and the other way around.

The **Neutralise** R package is available on a GitHub repository (<https://github.com/lucp9827/Neutralise>). An accompanying repository (<https://github.com/lucp9827/NeutraliseFiles>) contains a zip file with directories that contain all the statistical tests and simulation methods and scenarios that have already been submitted and evaluated (i.e., *neutralized*). All results of the corresponding simulations are available from here, as well as an annual report that summarizes the results according to the protocol (Appendix F). These repositories will be transferred to a repositories of the Data Science Institute (Hasselt University) upon acceptance of this paper. The GitHub page of Neutralise also contains a detailed manual that explains how the package can be installed and it includes a tutorial. In addition, the GitHub page of NeutraliseFiles contains two sets of concise step-by-step instructions for generating and/or analyzing the results from submitted methods and simulation scenarios (see also Appendices D and E).

The end user function of the R package can be run in two modes. The first mode (named *single*) allows a user to evaluate a single statistical method under a single simulation scenario. The second mode (named *all*) will evaluate a new statistical method under all scenarios available on GitHub, or evaluate all statistical methods available on GitHub under a new simulation scenario, or a combination of both. The user is asked to push the new results, which include the new R functions, to the GitHub repository (push request). The core team of this initiative will regularly check the consistency of the files on GitHub, and when needed the missing simulations will be run and the results will be pushed to GitHub.

Statistical tests can be added to the system as an R file with a single function, named *Test* and with one argument: *db*, which is a data frame with two columns. One column should be named *y* and contains the $n_1 + n_2$ sample observations, and the other column should be named *group* and contains values 1 and 2, referring to the groups the sample observations come from. The *Test* function should return the results (test statistic and p -value) as list with names *stat* and *p.value*. This R file must contain a header that gives a the name and a description of the method, and references to the literature if appropriate. An example is shown in Appendix A.1.

Simulation scenarios can also be added to the system as two R files. One file with a single function, named *Data.Generator* and with three arguments: (1) n_1 : size of sample from F_1 ; (2) n_2 : size of sample from F_2 ; (3) *parameters*: a vector with the numerical values of the parameters needed by the simulation procedure (e.g., the difference in means between F_1 and F_2 and the variances of these distributions). The file must also contain a header that gives a description of the simulation method. The result of this function should be a data frame with the same specifications as the *db* argument of the *Test* function. An example is shown in Appendix A.1. The second file contains a data frame, named *settings*, with the parameter values to be passed to the data generator function; each line of the data frame corresponds to another scenario. See again Appendix A.1 for an example. The combination of a data generator (i.e., distribution) and a parameter setting is referred to as a simulation scenario.

4.2 | Architecture

In this section, we provide some more details of the architecture of the R package and GitHub repository (see also Figure 1).

After installing the R package, the user must download the required directories from the repository on Github, which contains all methods and simulation scenarios that have already been run, as well as all the corresponding simulation results. In particular, five directories are copied: *Methods*, *Data*, *Settings*, *Issues*, and *Results*.

We will explain the architecture for the two operating modes *single* and *all*. For the former, the user has a single statistical test that needs to be evaluated with a single data generator with possibly multiple parameter settings. In this case, the user first creates (1) a function for the test (e.g., with name *My.Test*); (2) a function for the data generation (e.g., with name *My.Data.Generator*); and (3) a data frame that contains the settings (e.g., with name *my.settings*), that is, the parameter

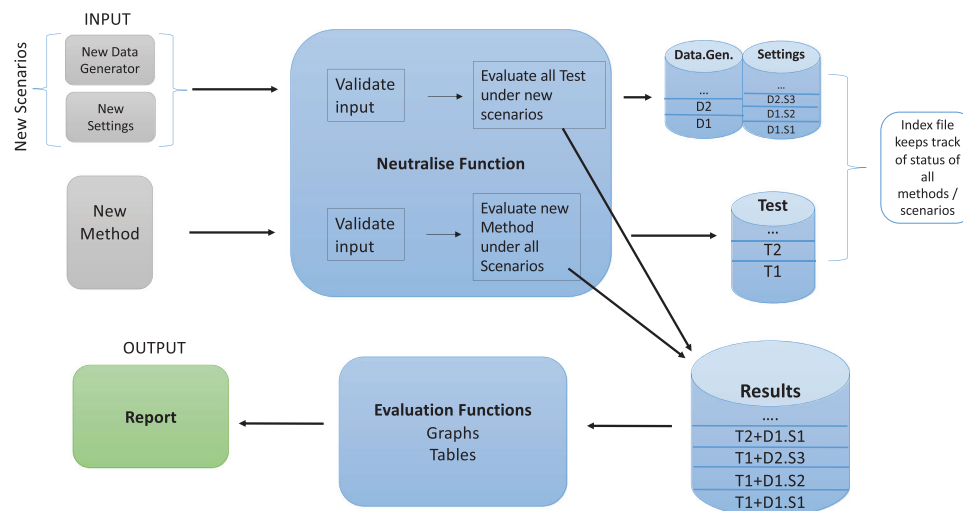


FIGURE 1 Schematic overview of the architecture of the R package.

values to be provided to the data generator. See Section 4.1 for more details on these three objects and Appendix A.1 for an example. The new method can be evaluated under the new simulation scenarios as follows:

```
> results<-Neutralise(path,test=My.Test, data.generator=My.Data.Generator,
settings=my.settings, N=10000)
```

where `path` refers to the parent directory to which the GitHub directory structure has been copied. The function will evaluate the test based on $N = 10,000$ Monte Carlo simulation runs in which data are generated with the function `Data.Generator` and with parameters set to those specified in `my.settings`. The results are written to a new subdirectory in the `Results/Local` folder. The following files are generated: (1) an R file with a function named `My.Test` (or other name given to the test function); (2) an R file with a function named `My.Data.Generator` (or other name given to the data generator function); (3) an R file with the data frame settings; (4) a text file with the R session information as generated by `sessionInfo()`; (5) a text file with the simulation results; and (6) a `a.RData` file with the simulation results. The results file contains all necessary information to reproduce the results, including the seed, and the results in terms of the power.

When running the function in *single* mode as shown above, the simulation results are also returned to the user in the data frame `results`. The *single* mode is designed for quick evaluation of a method under a limited number of simulation scenarios. In this way, it is also useful for statisticians who constructed a new two-sample test and want quick evaluation of their method in the development phase of their research. Note that this does not jeopardize the objectives of our neutral comparison initiative: once this researcher wants her/his method to be part of the project, it will be evaluated under all simulation scenarios. This can happen in the *all* mode.

In the operating mode *all*, the `Neutralise` function is used with either the argument `test`, or the arguments `data.generator` and `settings` set to `NULL`. For the argument that is set to `NULL`, the R function will look for R files in the corresponding directory. For example, if `test=NULL` and the other arguments are not set to `NULL`, then the simulation scenarios specified by the data generator function in the argument `data.generator` and the settings matrix in the argument `settings` are applied to all statistical test functions in the `Test_Finished` directory, and for each of these tests the simulation results are written to a new subdirectory that is created in the `Results` folder.

When the `Neutralise` function is called without providing a new test or simulation scenario (i.e., all these arguments are set to `NULL`), the function will look for new files in the directories `Methods`, `Data`, and `Settings` and will first do eligibility checks on the new functions and settings matrix. When these checks detect problems, these will be reported in a separate file that is saved in the `Issues` directory. When the new files pass the checks, the `Neutralise` function runs simulations for all combinations of tests and simulation scenarios, and results are written to the `Results` directory as explained earlier.

In the `Results` directory, there is also a file that contains the data frame `neutralise.status`, which keeps track of all methods, data generators and settings files, and their status: for example, did the file pass the checks, is the method neutralized, and so forth.

So far we have seen that detailed results are written to files in the `Results` directory. Summarizing results in a fair and unbiased fashion is also an important part of neutral comparison studies. In Section 5, this is discussed and illustrated.

4.3 | Maintenance

Just like any other R package or project, the code and the GitHub repository must be maintained. In this section, we discuss some relevant aspects: R package, completeness and availability of the results, and the public repository.

The initial version of the R package (Version 0.1.0) contains R functions for the Neutralise function and functions to summarize the simulation results in plots. The GitHub repository contains 15 tests and eight data generators with settings for the scenarios. The following tests are included; Anderson–Darling (AD; Scholz & Stephens, 1987), Brunner–Munzel (Brunner & Munzel, 2000), Baumgartner–Weiss–Schindler (Baumgartner et al., 1998), Cramer–Von Mises (Brown, 1982), the Percentile Modified WMW test of Gastwirth (Gastwirth, 1965), Hogg–Fisher–Randles adaptive test (Hogg et al., 1975), Kolmogorov–Smirnov (KS; Smirnov, 1939), Mood’s median test (Mood, 1954), Asymptotic smooth test with fixed order 4 (and 6) and Legendre polynomials (Janic-Wróblewska & Ledwina, 2000), two-sample Student- t test, the Welch t -test (Welch, 1938), van der Waerden normal scores test (van der Waerden, 1952), Asymptotic WMW test (Wilcoxon, 1992), and Yuen’s test (Yuen, 1974). In addition, the Github repository contains the simulation results for all combinations of these tests and simulation scenarios. At this stage, the GitHub repository is *complete* in the sense that it contains all results for all the statistical tests and scenarios included.

Once users start using the R package for evaluating their own tests, or for evaluating existing tests under new scenarios, we (the core team) lose control about the availability of the results and hence also about the completeness. Although we ask all users to contribute to our project and to the scientific community by sending us their R functions for new tests and simulation scenarios (via a push request on GitHub), we must be realistic that not everyone will do so. We believe, however, that everyone will benefit from sharing code and results, particularly if this project gains traction and will be consulted by referees and peers (see the discussion in Section 6) so that researchers who want to publish new statistical tests will be asked to participate in this project.

5 | SOME RESULTS

With the simulation results on GitHub, everyone is free to do further analysis and draw conclusions. In the R package, there are already a few functions available to make summary graphs. These functions are also available in an R Shiny app. The app can be downloaded from the GitHub repository for running on a local computer, but it can also be directly run from our Shiny app server (<https://dsi-uhasselt.shinyapps.io/Neutralise/>). In this section, we illustrate the use of these functions, based on a small set of simulation results for the following tests and data generators (note that the GitHub repository includes more tests than those we focus on here). Tests: WMW test (asymptotic normal approximation), the Welch t -test, the KS test, and the AD test. Data generators: normal with equal variances, normal with unequal variances, logistic with equal variances, exponential distribution, Cauchy distribution, and the generalized Tukey Lambda distribution. The latter distribution is very flexible for simulating distributions with many different shapes. Here, we have used this generalized Lambda distribution (GLD) for generating skew distributions and only LS alternatives are considered; it is abbreviated as GLDLS. See Appendix B for more details on the quantile function of the distribution. Another flexible distribution that has been included here, is the g -and- h distribution (Tukey, 1977) with equal variances (ghEqual) and with a specified kurtosis parameter (ghEqualK); See Appendix C for more details on the quantile function of the distribution.

We will illustrate three types of graphs: (1) boxplots for assessing type I error rate control; (2) power curves for individual tests; and (3) power–power plots for comparing two or more tests.

5.1 | Boxplots for assessing type I error rate control

Figure 2 shows boxplots of the empirical type I error rates of the Welch t -test, WMW test, the KS test, and the AD test at the nominal level of 5%, for the distributions selected here. A reference line indicates the largest acceptable estimated type I error rate, that is, tests for which the type I error rates are estimated above this line are liberal (this line is based on a 95% confidence interval). Similarly, another reference line indicates the smallest acceptable estimate; points below this line correspond to conservative tests. From these graphs, we see that the KS test is conservative for most of the distributions, whereas the WMW and the AD tests have overall good type I error rate control. The Welch t -test has good control, except for the Cauchy distribution and for some of the scenarios with the g -and- h distribution and with the exponential distribution. The scenarios with the g -and- h distribution and the exponential distribution that do not control the type I error rate, are

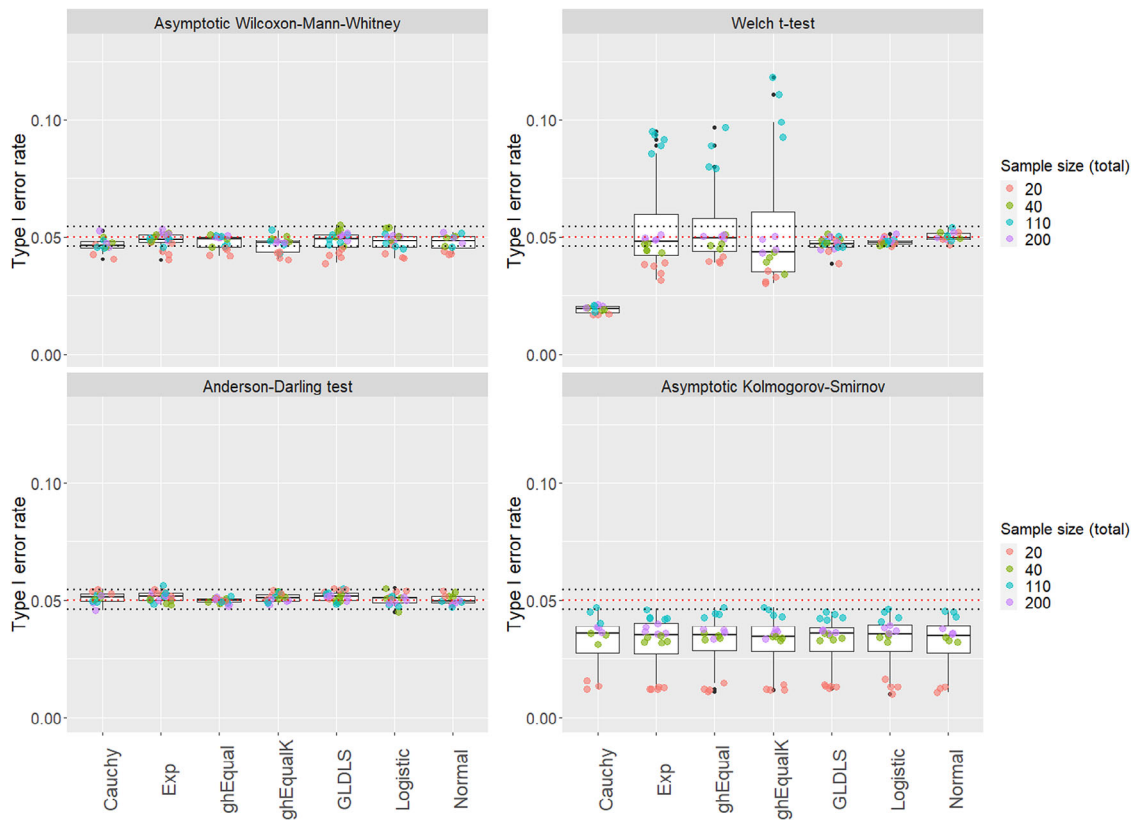


FIGURE 2 Boxplots of the empirical type I error rates of the Wilcoxon–Mann–Whitney (WMW) test (top, left), Welch t -test (top, right), Anderson–Darling (AD) test (bottom, left), and Kolmogorov–Smirnov (KS) (bottom, right) for the different data-generating distributions. The color of the points refer to the total sample size. All tests were performed at the 5% level of significance (red dotted line), and the error rates were computed based on 10,000 simulation runs. Points above (below) the lower and upper horizontal reference lines (black dotted line) correspond to liberal (conservative) tests.

scenarios with an unbalanced design. In addition, this result agrees with the expectations from the literature (Skovlund & Fenstad, 2001), as the mentioned distributions can be skewed and show heavier tails, and the Welch t -test behaves best for symmetric distributions with moderate tails. The information about the moments of the distributions can be easily found by using the interactive brushing function in the Shiny app.

5.2 | Power curves for individual tests

The empirical powers can be plotted versus the difference in means of the data-generating distributions. These are very common plots for summarizing the results of a simulation study. Figure 3 shows power curves for the Welch t -test for several sample sizes scenarios with the normal distribution. The R function allows to filter out test/scenarios combinations that do not control the type I error rate; this allows for a fair comparison. In the Shiny app, an interactive brushing function is available to get more detailed information for the individual scenarios (points in the plot).

5.3 | Power–power plots

Because the objective of this project is to generate results for many tests and many scenarios, much more than what it is typically done for a simulation study section in a paper, presenting all results as power curves becomes almost infeasible, or at least these plots no longer succeed well in summarizing the results. We therefore propose two other types of plots. The first exist in plotting the power of one test against the power of another test; each point corresponds to a single scenario. In this way, two tests can be compared for all scenarios in a single plot. Just like before, the R function allows for filtering

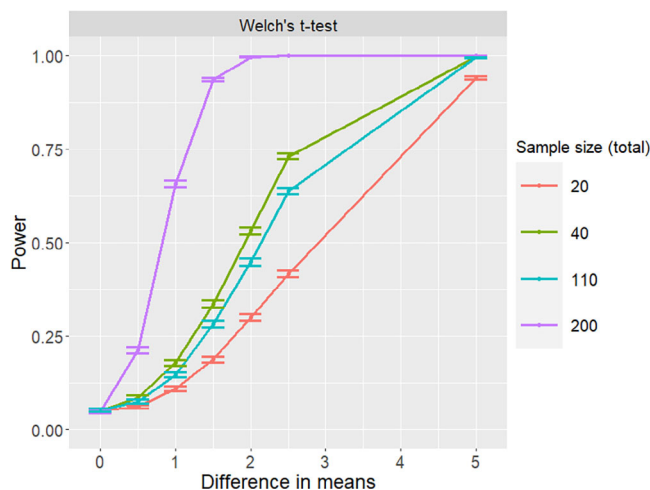


FIGURE 3 Power curves for several sample sizes for the Welch t -test for the normal distribution (standard deviation equal to 3). The empirical powers for the test at the 5% level of significance, are plotted against the difference in means. The 95% confidence intervals of the powers are also indicated. Tests/scenario combinations that do not control the type I error rate, were filtered out.

our scenarios for which one of the tests has no type I error rate control, or for selecting a subset of the scenarios (e.g., only scenarios for the normal distribution). Since the plot is constructed by plotting two powers against one another, it is referred to as a power–power plot. Figure 4 shows three power–power plots. In the top left panel, the powers of the WMW test are compared to those of the Welch t -test, for all scenarios and all sample sizes. The plot shows that many points are close to the diagonal (particularly for the symmetric distributions), indicating that for these distributions the powers of the two tests are close to one another. However, there are still many other points are far off the diagonal, toward both sides. This shows that for some scenarios (skew and/or heavy-tailed distributions) the powers of the WMW test and the Welch t -test may be very different. Sometimes the WMW has higher power, other times the Welch t -test has higher power. The textual output of the function gives a very short summary:

Normal2Var:

WMW_Asymp has higher power than TTest_VarUnequal in 23.6% of the 72 scenarios where both methods control the type I error rate.

Normal:

WMW_Asymp has higher power than TTest_VarUnequal in 22.2% of the 72 scenarios where both methods control the type I error rate.

Exp:

WMW_Asymp has higher power than TTest_VarUnequal in 75.8% of the 91 scenarios where both methods control the type I error rate.

Cauchy:

WMW_Asymp has higher power than TTest_VarUnequal in 99.3% of the 144 scenarios where both methods control the type I error rate.

ghEqual:

WMW_Asymp has higher power than TTest_VarUnequal in 47.6% of the 105 scenarios where both methods control the type I error rate.

ghEqualK:

WMW_Asymp has higher power than TTest_VarUnequal in 53.3% of the 105 scenarios where both methods control the type I error rate.

GLDLs:

WMW_Asymp has higher power than TTest_VarUnequal in 61.3% of the 75 scenarios where both methods control the type I error rate.

Logistic:

WMW_Asymp has higher power than TTest_VarUnequal in 61.6% of the 112 scenarios where both methods control the type I error rate.

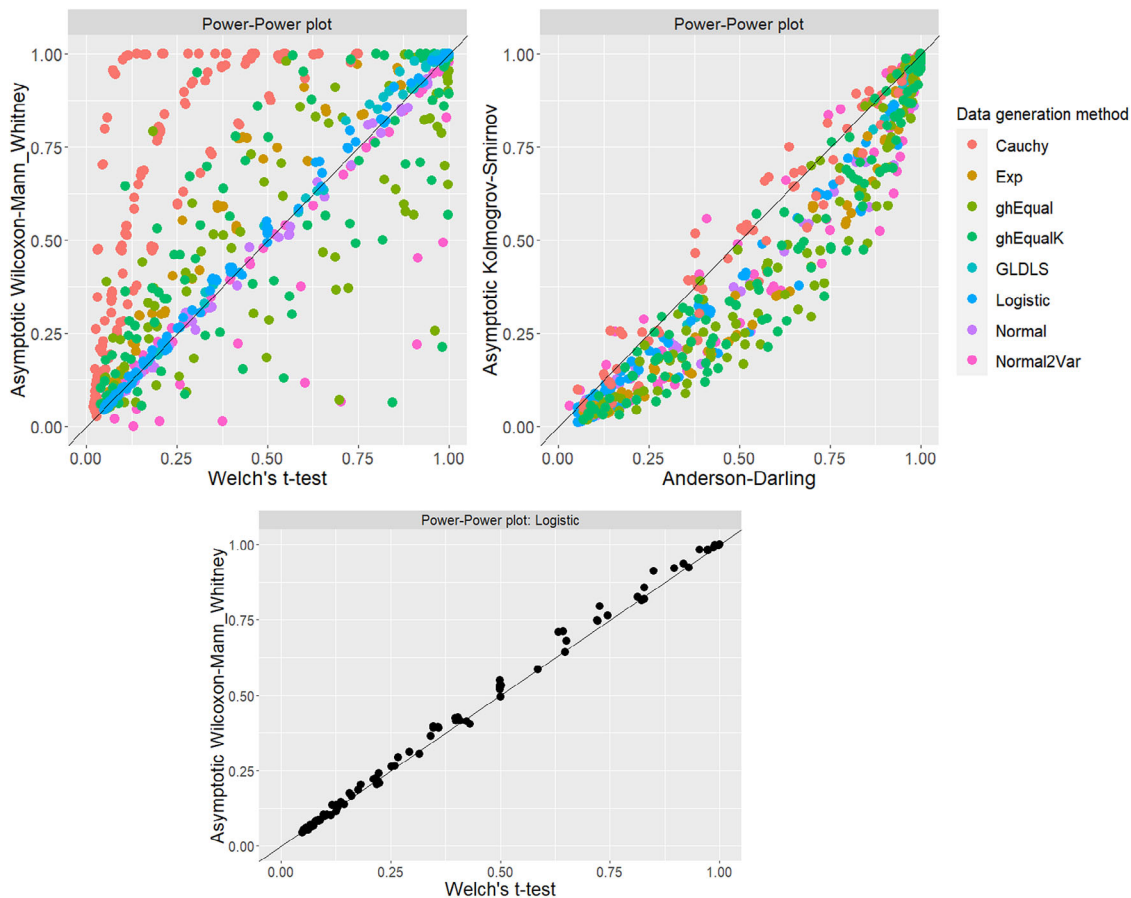


FIGURE 4 Power–power plots for comparing two tests. Top Left: Empirical powers of the Welch t -test versus the Wilcoxon–Mann–Whitney (WMW) test for all scenarios. Of the 900 scenarios tested, 125 were filtered out due to at least one method not controlling the type I error rate. Top right: Kolmogorov–Smirnov (KS) test versus Anderson–Darling (AD) test. Of the 900 scenarios, 87 were filtered out due to at least one method not controlling the type I error rate. Bottom: Welch t -test versus the WMW, but only for scenarios with the logistic distribution. Both methods controlled the type I error rate for all logistic scenarios. All tests were performed at the 5% level of significance, and the error rates were computed based on 10,000 simulation runs. Tests/scenario combinations that do not control the type I error rate were filtered out.

A similar plot is shown in the bottom panel of Figure 4, but now limited to scenarios with the logistic distribution, for which we know that the WMW test is the locally most powerful rank test and hence we expect to WMW test to have higher power than the Welch t -test. For the logistic distribution, the power–power plot shows that (1) the powers of the tests are always similar (points are close to the diagonal), and (2) the WMW often wins over the Welch t -test. Note that when the Welch t -test outperforms the WMW test, it is only by a very small margin. This may be explained by the (small) imprecision with which the powers are calculated in the simulation study (based on 10,000 simulation runs). The textual output of the R function now reads as

Logistic:

```
WMW_Asymp has higher power than TTest_VarUnequal in 61.6%
of the 112 scenarios where both methods control the type I error rate.
```

Finally, the third panel (top right) in Figure 4 compares the KS and AD tests, showing that the AD test outperforms the KS test more often than the other way around. Moreover, the asymmetry of the plot suggests that when the AD outperforms the KS test, it is by a larger margin than it is the other way around. Note that since the KS test is very conservative in the type I error rate, which results in low power, the KS test was not the best choice to compare to AD. We have chosen it here, however, as a very obvious example for illustrating the interpretation of this type of plot.

A second type of power–power plot consists in plotting the power of one test against the highest power of the competitor tests (after filtering out test/scenario combinations with poor type I error rate control). This allows for focusing on a single

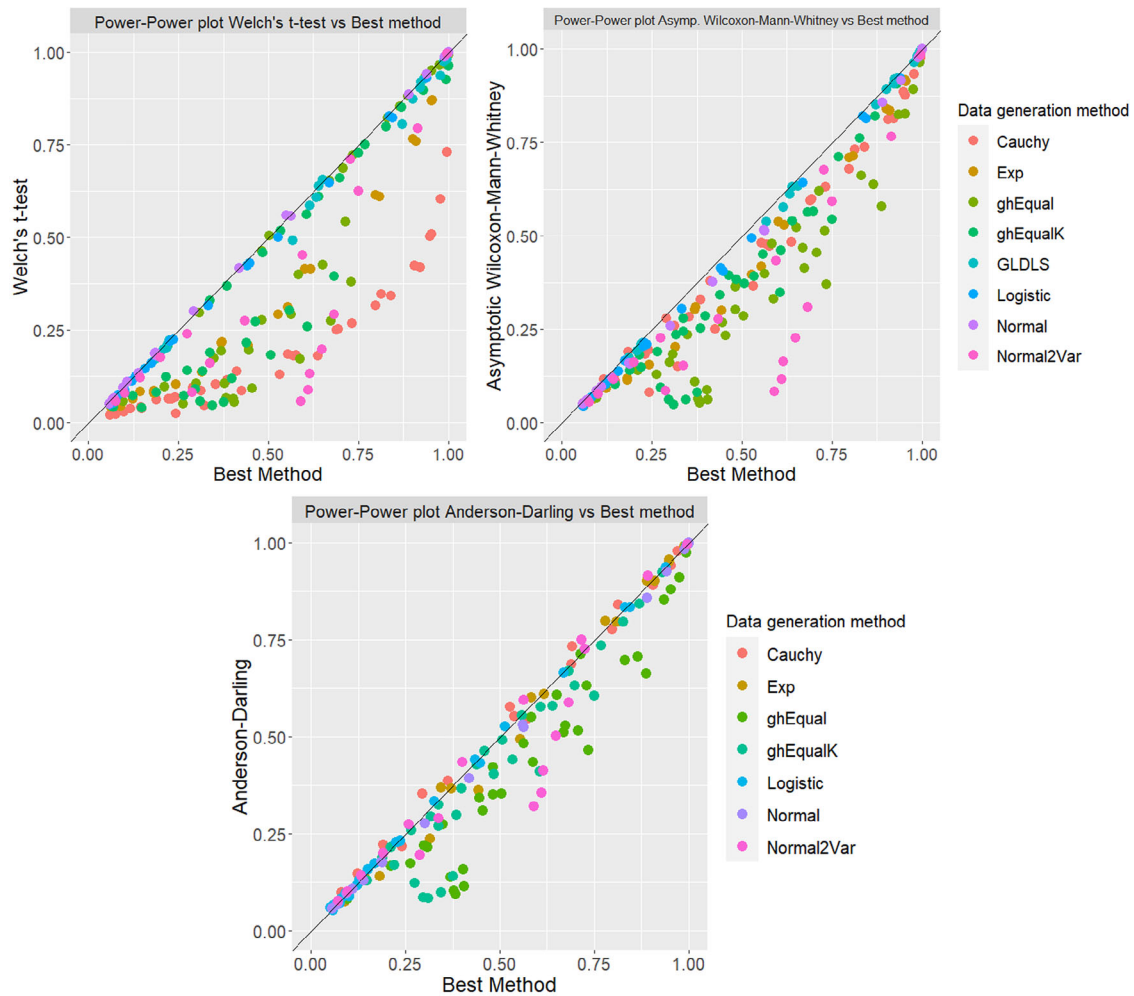


FIGURE 5 Power–power plots for comparing a single test with the best competitor. Top left: Welch t -test. Top right: Wilcoxon–Mann–Whitney (WMW) test. Bottom: Anderson–Darling (AD) test. All tests were performed at the 5% level of significance, and the powers were computed based on 10,000 simulation runs. Only results for a total sample size of 20 are shown. Tests/scenario combinations that do not control the type I error rate were filtered out.

test, while still involving all other tests. Figure 5 shows three such plots for simulation results for a total sample size of 20. For the Welch t -test, the graph shows that the test has the largest power for the scenarios with the normal distribution with equal variances, but for other distributions other tests have larger powers. Remarkably, for normal distributions with unequal variances, the Welch t -test has much lower power than the best competitor. For the WMW test, the power–power plot shows that the test has the largest power for the logistic distribution (as expected), but for other scenarios better tests exist. Upon comparing the power–power plots of the Welch t -test and the WMW test, we see that there are less distributions included in former. This is a consequence of filtering out scenarios for which the Welch t -test does not control the type I error rate (see also Figure 2). The power–power plot for the AD test shows that the AD test frequently has the largest power, and when it does not, its power is not much smaller than the power of the best competitor (i.e., the points are close to the diagonal).

6 | CONCLUSION

Neutral comparison of statistical tests is an important aspect of scientific integrity and reproducibility in statistical science. Despite the huge number of simulation studies that have been published in the scientific literature, none of them is complete in the sense that all competitors are included and that tests are evaluated under a huge number of scenarios.

Moreover, we have argued that the conclusions from these studies may be biased by the choices of simulation scenarios and competitor tests by the authors.

In this paper, we have described a new open science initiative for neutral comparisons of statistical tests for the very simple two-sample problem. It is set up as an open-source R package (Neutralise), an R Shiny app, and a GitHub repository. The latter does not only contain the R package, but it also contains a manual and a demo, as well as the simulation results. Every user can run the Neutralise function on new statistical tests under new simulation scenarios, but, more importantly, new tests can also be evaluated under all simulation scenarios that have been submitted before by any other scientist participating in this initiative. In this way, a test can be *neutralized*. Similarly, everyone can add a new simulation scenario to the system, and all statistical tests that have been submitted before, will be evaluated. The participants of our initiative are invited to submit the new simulation results to the GitHub repository. In this way, a large collection of results can be gathered, with minimal risk of introducing bias. All code and results are publicly available and can be downloaded by anyone who wants a fair comparison of statistical procedures.

The R package comes with functions that can assist in summarizing the simulation results, and make informative summary graphs. However, as our initiative is meant to become an open science collaborative project, we invite statisticians to contribute not only methods and simulation scenarios, but also build new functions for summarizing and exploring the results. The functions for exploring the simulation results are also available in an easy-to-use R Shiny implementation (<https://dsi-uhasselt.shinyapps.io/Neutralise/>).

This paper was not meant to present results of a neutral comparison study. It rather describes the architecture of an open science initiative, and it demonstrates how methods can be *neutralized*. We have deliberately chosen for a very simple statistical problem. If our approach turns out to be successful, we hope that it can inspire statisticians to setup similar initiatives for other statistical procedures. For example, in the context of genomics and microbiome studies, many methods have been proposed for testing for differential gene expression or differential abundance. Although these methods are essentially also two-sample tests, the multiple testing issue brings along an extra level of complication, as well as the need for simulating realistic data that cannot be simply described by any of the traditional distributions (Assefa et al., 2020; Hawinkel et al., 2020).

We hope that many scientist will contribute to this open science project so that eventually a comprehensive neutral comparison of two-sample tests will become available at all times. One could even envision a scientific community in which journal editors require their authors to evaluate their new methods in open science initiatives as ours. International professional organizations within the statistics discipline may even take a lead in hosting such initiatives.

ACKNOWLEDGMENTS

The first author wants to thank Geert Jan Bex for inspiring discussions on cybersecurity and on the use of GitHub, and Arne Bathke for directing us to the special issue of the *Biometrical Journal*. This project is partially funded by the Hasselt University Grand Challenge BOF project BOF21GP17.


CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available in the supplementary material of this article.

OPEN RESEARCH BADGES

 This article has earned an Open Data badge for making publicly available the digitally-shareable data necessary to reproduce the reported results. The data is available in the [Supporting Information](#) section.

This article has earned an open data badge “**Reproducible Research**” for making publicly available the code necessary to reproduce the reported results. The results reported in this article could fully be reproduced.

ORCID

Leyla Kodalci  <https://orcid.org/0000-0003-2804-2547>

REFERENCES

- Assefa, A. T., Vandesompele, J., & Thas, O. (2020). SpsimSeq: Semi-parametric simulation of bulk and single-cell RNA-sequencing data. *Bioinformatics*, 36(10), 3276–3278.
- Baumgartner, W., Weiß, P., & Schindler, H. (1998). A nonparametric test for the general two-sample problem. *Biometrics*, 54(3), 1129–1135.
- Boulesteix, A.-L., Lauer, S., & Eugster, M. J. (2013). A plea for neutral comparison studies in computational sciences. *PLoS One*, 8(4), e61562.
- Brown, B. M. (1982). Cramer-Von Mises distributions and permutation tests. *Biometrika*, 69(3), 619–624.
- Brunner, E., & Munzel, U. (2000). The nonparametric Behrens-Fisher problem: Asymptotic theory and a small-sample approximation. *Biometrical Journal*, 42(1), 17–25.
- Freimer, M., Kollia, G., Mudholkar, G. S., & Lin, C. T. (1988). A study of the generalized Tukey Lambda family. *Communications in Statistics-Theory and Methods*, 17(10), 3547–3567.
- Gastwirth, J. L. (1965). Percentile modifications of two sample rank tests. *Journal of the American Statistical Association*, 60(312), 1127–1141.
- Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D.-H., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang, X., Athanasakis, D., Shave-Taylor, J., Milakov, M., Park, J., ... Bengio, Y. (2013). Challenges in representation learning: A report on three machine learning contests. In *Proceedings of Neural Information Processing: 20th International Conference, ICONIP 2013, Daegu, Korea, November 3-7, Part III 20* (pp. 117–124). Springer.
- Hawinkel, S., Rayner, J., Bijmens, L., & Thas, O. (2020). Sequence count data are poorly fit by the negative binomial distribution. *PLoS One*, 15(4), e0224909.
- Hoffmann, F., Bertram, T., Mikut, R., Reischl, M., & Nelles, O. (2019). Benchmarking in classification and regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(5), e1318.
- Hogg, R. V., Fisher, D. M., & Randles, R. H. (1975). A two-sample adaptive distribution-free test. *Journal of the American Statistical Association*, 70(351a), 656–661.
- Hyndman, R. J. (2020). A brief history of forecasting competitions. *International Journal of Forecasting*, 36(1), 7–14.
- Janic-Wróblewska, A., & Ledwina, T. (2000). Data driven rank test for two-sample problem. *Scandinavian Journal of Statistics*, 27(2), 281–297.
- Mood, A. M. (1954). On the asymptotic efficiency of certain nonparametric two-sample tests. *Annals of Mathematical Statistics*, 25(2), 514–522.
- Prangle, D. (2017). gk: An R package for the g-and-k and generalised g-and-h distributions. *arXiv preprint arXiv:1706.06889*.
- Scholz, F. W., & Stephens, M. A. (1987). K-sample Anderson–Darling tests. *Journal of the American Statistical Association*, 82(399), 918–924.
- Skovlund, E., & Fenstad, G. U. (2001). Should we always choose a nonparametric test when comparing two apparently nonnormal distributions? *Journal of Clinical Epidemiology*, 54(1), 86–92.
- Smirnov, N. (1939). Sur les écarts de la courbe de distribution empirique. *Matematicheskii Sbornik*, 48(1), 3–26.
- Tukey, J. W. (1977). Modern techniques in data analysis. In *Proceedings of the NSF-Sponsored Regional Research Conference* (Vol. 7). Southern Massachusetts University North Dartmouth.
- van der Waerden, B. (1952). Order tests for the two-sample problem and their power. In *Indagationes Mathematicae (Proceedings)* (Vol. 55, pp. 453–458). Elsevier.
- Weber, L. M., Saelens, W., Cannoodt, R., Sonesson, C., Hapfelmeier, A., Gardner, P. P., Boulesteix, A.-L., Saeys, Y., & Robinson, M. D. (2019). Essential guidelines for computational method benchmarking. *Genome Biology*, 20(1), 1–12.
- Welch, B. L. (1938). The significance of the difference between two means when the population variances are unequal. *Biometrika*, 29(3/4), 350–362.
- Wilcoxon, F. (1992). *Individual comparisons by ranking methods*. Springer.
- Yuen, K. K. (1974). The two-sample trimmed *t* for unequal population variances. *Biometrika*, 61(1), 165–170.

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

How to cite this article: Kodalci, L., & Thas, O. (2024). Neutralise: An open science initiative for neutral comparison of two-sample tests. *Biometrical Journal*, 66, 2200237. <https://doi.org/10.1002/bimj.202200237>

APPENDIX A

A.1 | Example R files

Example of an R file with a test function:

```
# NAME
# Asymptotic Wilcoxon-Mann-Whitney test
# DESCRIPTION
```

```
# Two sample Wilcoxon rank sum test = Mann-Whitney test. P-values
# based on asymptotic approximation
# REFERENCES
# Wilcoxon, F. (1945). Individual comparisons by ranking methods.
# Biom. Bull., 1, 80-83.
# END

WMW_Asymp<-function(db) {
  results<-wilcox.test(y~group,data = db, exact=FALSE)
  return(list(
    stat=results$statistic,
    p.value=results$p.value
  ))
}
```

Example of an R file with a data generator function:

```
# DESCRIPTION
# simulation of two normal distributions with unequal variance
# END

Normal2Var<-function(n1=10,n2=10,parameters=c(0,1,2)) {
  delta<-parameters[1]
  sd1<-parameters[2]
  sd2<-parameters[3]
  y<-c(rnorm(n1,sd=sd1),
  rnorm(n2,mean = delta,sd=sd2))
  db<-data.frame(y=y, group=rep(c(1,2),c(n1,n2)))
  return(db)
}
```

Example of an R file with some settings for the Data.Generator function:

```
settings<-data.frame(
  delta=c(0,0,1,1.5,2,2.5),
  sd1=c(1,1,1,1,1,1),
  sd2=c(1,2,2,2,2,2),
  null=c(1,0,0,0,0,0)
)
```

Example of the Neutralise function that uses the inputs that are described above (in the single mode). Note that all the inputs should be loaded in the R-environment you are working in.

```
> results<-Neutralise(path,test=WMW_Asymp, data.generator=Normal2Var,
  settings=settings, N=10000)
```

A.2 | Example output files

The example output file is a snippet of the results of Kolmogorov–Smirnov (KS) test with generalized Lambda distribution with location shift (GLDLS) data generation method. All the results are provided in the NeutraliseFiles repository. See Appendix B, for more details on the parameters for the GLDLS distribution.

```
"method", "distribution", "seed", "N", "n1", "n2",
"delta", "scale", "lambda3", "lambda4",
"null", "power0.01", "l_CI", "u_CI",
"power0.05", "l_CI", "u_CI",
"power0.10", "l_CI", "u_CI",
"ct_0.10", "ct_0.05", "ct_0.01"
```

```

"KS", "GLDLS", 7665848, 10000, 10, 10,
0, 2, -0.002, -0.002,
1, 0.0017, 0.0006, 0.0028,
0.0122, 0.0101, 0.0144,
0.0549, 0.0512, 0.0587,
549, 122, 17
"KS", "GLDLS", 916321, 10000, 10, 10,
0, 2, -0.02, -0.02,
1, 0.0019, 0.0008, 0.0030,
0.0134, 0.0112, 0.0157,
0.0527, 0.0490, 0.0564,
527, 134, 19
"KS", "GLDLS", 59113046, 10000, 10, 10,
0, 2, -0.05, -0.05,
1, 0.0021, 0.0009, 0.0033,
0.0141, 0.0118, 0.0164,
0.053, 0.0492, 0.0566,
529, 141, 21

```

In general, the output files (text or RData-file) contain the following elements:

- method = method name
- distribution = data generation method²
- seed = seed number to reproduce the exact results
- N = number of simulation runs
- n1 and n2 = sample size per group
- delta = location shift between the two groups
- ... parameters that are specific for the data generation method
- null = 1 if the scenario is under H_0 , which is used as indicator variable.
- power0.01 = power results with significance level of 1% + l_CI,U_CI which represent the lower and upper confidence limits.
- power0.05 = power results with significance level of 5% + l_CI,U_CI which represent the lower and upper confidence limits.
- power0.10 = power results with significance level of 10% + l_CI,U_CI which represent the lower and upper confidence limits.
- ct_0.10 = the number of times H_0 gets rejected in the simulation runs with 10% significance level
- ct_0.05 = the number of times H_0 gets rejected in the simulation runs with 5% significance level
- ct_0.01 = the number of times H_0 gets rejected in the simulation runs with 1% significance level

APPENDIX B: THE GENERALIZED TUKEY LAMBDA DISTRIBUTION

The generalized Tukey Lambda distribution has many parameterizations. We have used the parameterization of Freimer et al. (1988). The quantile function is given by

$$Q(p) = \lambda_1 + \frac{1}{\lambda_2} \left[\frac{p^{\lambda_3} - 1}{\lambda_3} - \frac{(1-p)^{\lambda_4} - 1}{\lambda_4} \right]$$

for $p \in [0, 1]$. The parameters λ_1 and λ_2 are location and scale parameters, respectively, and the parameters λ_3 and λ_4 are shape (or tail) parameters. The distribution is defined for $\lambda_1, \lambda_3, \lambda_4 \in \mathbb{R}$ and $\lambda_2 \in \mathbb{R}^+$.

For particular choices of $\lambda_3 = \lambda_4$, the distribution approximates the normal distribution very well. This class of distributions include many different shapes: symmetric and left- and right-skewed distributions, heavy- and light-tailed distributions, and finite and infinite support distributions. Simulation from this distribution is very simple as it is defined in terms of its quantile function.

APPENDIX C: THE g -AND- h DISTRIBUTION

The generalized g -and- h distribution (Prangle, 2017; Tukey, 1977) is defined by the following quantile function:

$$Q(p) = A + B \left[1 + c \tanh \left(\frac{gz_p}{2} \right) \right] \exp \left(\frac{hz_p^2}{2} \right),$$

where $z_p = \Phi^{-1}(p)$ is the quantile of the standard normal distribution corresponding to p , parameters A and B are location and scale parameters, respectively, and c is a constant (fixed at 0.8). Parameter g measures the skewness of the distribution, and parameter h is related to the kurtosis. Parameter restrictions exist in $B > 0$ and $h \geq 0$.

The sign of parameter g indicates the direction of skewness. When $g = 0$ the distribution is symmetric. Increasing the parameter h , adds more kurtosis to the distribution, and when $h = 0$ the kurtosis agrees with the standard normal distribution. Note that due to restriction on h , distributions with less kurtosis than the normal distribution cannot be simulated by the g -and- h distribution. However, these can be simulated with the generalized Tukey Lambda distribution (Appendix B).

APPENDIX D: STEP-BY-STEP INSTRUCTIONS FOR NEUTRALISE: STARTING FROM AN EMPTY NEUTRALISEFILES FOLDER

This concise checklist is intended for users to install Neutralise and start working from the beginning, an empty NeutraliseFiles folder, and generate their own results. For more details and examples, we refer to the README file of Neutralise (<https://github.com/lucp9827/Neutralise>). We recommend following the tutorial to understand how the Neutralise function works and what the possibilities are within this initiative.

1. Download NeutraliseFiles_empty.zip from <https://github.com/lucp9827/NeutraliseFiles> and **unzip** this folder.
2. Start R.
3. Set the working directory of your R-session to the folder NeutraliseFiles in the unzipped NeutraliseFiles_empty folder.
4. Save this working directory also to the object path. (R-code: `path=getwd()`)
5. Install and load the following packages in your R-session: `remotes`, `kSamples`, `lawstat`, `BWStest`, `twosamples`, `RVAideMemoire`, `WRS2`, `gk`, `gld`, and `DescTools`.
Rcode to Install & load required packages:

```
reqpkg = c("remotes","kSamples","lawstat","BWStest","RVAideMemoire","WRS2",
"gk","gld","twosamples","DescTools")
for(i in reqpkg)
{print(i)
install.packages(i)
library(i, quietly=TRUE, verbose=FALSE,
warn.conflicts=FALSE, character.only=TRUE)}
```
6. Install Neutralise. You can use the following R-code: `remotes::install_github('lucp9827/Neutralise')`.
7. Load Neutralise in your R-session. (R-code: `library(Neutralise)`)
8. Run the function `Initialise_Neutralise(path)`
9. You can start using the Neutralise function. (Check the demonstration file for details on the function possibilities)

APPENDIX E: STEP-BY-STEP INSTRUCTIONS FOR NEUTRALISE: STARTING FROM THE GITHUB REPOSITORY NEUTRALISEFILES

This concise checklist is intended for users to analyze the results from the GitHub repository NeutraliseFiles. For more details and examples, we refer to the README file of Neutralise (<https://github.com/lucp9827/Neutralise>). We recommend following the tutorial to understand how the Neutralise function works and what the possibilities are within this initiative.

1. Follow steps 1 to 7 of the Step-by-Step Instructions in Appendix D, but in step 1 download the NeutraliseFiles_full.zip (instead of empty), and in step 3 set the working directory to the NeutraliseFiles folder in the unzipped NeutraliseFiles_full file (instead of empty).

2. You are ready to start analyzing the data, we propose two options below. However, if you want to start from the original result files, check the Demonstration file to understand how the results are organized in the NeutraliseFiles.
Option 1: You can start analyzing the data with the provided visualization functions (power–power plot, type I error rate boxplots, power curves), check the demonstration file in the Neutralise package for more details.
Option 2: You can use the summarized.RData files of all the results (type I error rate, power) for your own code or analysis. The files without `_df` extension are lists per data generation method and include all the setting parameters per scenario and the results, while the files with `_df` extension are data frames with an id number for the setting and the results (without setting parameters).

APPENDIX F: PROTOCOL FOR THE ANNUAL SUMMARY REPORT

This document describes the protocol that will be followed for producing the annual summary report.

1. Information:
 - Provide an overview of the tests and the scenarios included for evaluation. This can be presented in an appendix.
 - Provide information about the package version.
2. Type I error rate:
 - Screen tests under H_0 scenarios and report about controlling the nominal type I error rate. The results will be presented as boxplots of type I error rates: one boxplot per test/distribution combination. Jittered points are added to the boxplot, with colors referring to the sample size.
 - Tests that have a type I error rate larger than the nominal 5% level of significance, will not be further evaluated (95% confidence limits accounting for the Monte Carlo sampling variability will be used for this purpose).
3. Power
 - For each test that controls the type I error rate, power–power curves for comparison with the best test are presented, separately for $n = 20$ and $n = 200$. Each power–power plot is complemented with a numerical summary: the proportion of scenarios for which the test has larger power than the best test, and the median of the power differences for scenarios under which the test has smaller power than the best test.
 - Based on the previous set of graphs, a few overall good tests will be selected, and these tests are two-by-two compared with power-power plots.
 - Based on the two previous sets of graphs, and using information on the moments of the distributions, we try to understand under what scenarios tests perform better or worse than others.