Statistics and Computing (2023) 33:110
https://doi.org/10.1007/s11222-023-10271-x

ORIGINAL PAPER

euMMD: efficiently computing the MMD two-sample test statistic for univariate data

Dean A. Bodenham[1] · Yoshinobu Kawahara[2,3]

Received: 23 August 2022 / Accepted: 22 June 2023 / Published online: 27 July 2023

Abstract
The maximum mean discrepancy (MMD) test is a nonparametric kernelised two-sample test that, when using a characteristic kernel, can detect any distributional change between two samples. However, when the total number of d-dimensional observations is n, direct computation of the test statistic is $\mathcal{O}(dn^2)$. While approximations with lower computational complexity are known, more efficient methods for computing the exact test statistic are unknown. This paper provides an exact method for computing the MMD test statistic for the univariate case in $\mathcal{O}(n \log n)$ using the Laplacian kernel. Furthermore, this exact method is extended to an approximate method for d-dimensional real-valued data also with complexity log-linear in the number of observations. Experiments show that this approximate method can have good statistical performance when compared to the exact test, particularly in cases where $d > n$.

Keywords Two-sample testing · MMD · Univariate

1 Introduction

Two-sample testing is important in many areas of science and commerce and has been extensively studied in the statistics literature, particularly in the univariate case. The *maximum mean discrepancy* (MMD) test (Borgwardt et al. 2006; Gretton et al. 2012a) is a nonparametric kernelised two-sample test that, when using a characteristic kernel (Sriperumbudur et al. 2010; Fukumizu et al. 2009) on data in $\mathbb{R}^d$, can detect any change in a distribution between the two samples (Fukumizu et al. 2004, 2008). An expression for the MMD test statistic is given in Eq. (1) in Sect. 2, from which it is apparent that the computational complexity of computing the MMD statistic for two samples containing a total of n observations from d-dimensional data is $\mathcal{O}(dn^2)$. When the total number of observations n is large, the cost of computing the MMD can be therefore be prohibitively expensive. There have been several proposed approaches for speeding up the computation of the MMD statistic (Gretton et al. 2012a; Zaremba et al. 2013; Zhao and Meng 2015); note however that these are all approximate and have worse statistical performance.

In this paper, we derive an *exact* method for computing the MMD statistic in $\mathcal{O}(n \log n)$ for the case where the n observations are univariate. To do so, we use the Laplacian kernel, which is a popular characteristic kernel (Fukumizu et al. 2009; Sriperumbudur et al. 2010). This novel algorithm is named `euMMD`, for *e*fficient *u*nivariate *M*aximum *M*ean *D*iscrepancy.

There are two cases where such a computational speed up would have utility. The first is when an analysis involves a single test and the number of observations n is large; then the improvement in speed is clear as shown in Fig. 3 and Table 1. A second, less obvious, case is when an analysis requires many tests even though the number of observations is of moderate size; for example, in cyber-security applications the same two-sample test would be applied a large number of times across a population of entities (Neil et al. 2013), and in the medical literature there are situations where sample-size calculations are done via simulations which compute the test statistic for many different samples (Landau and Stahl 2013). In such cases, moderate gains in efficiency for each test would lead to a large overall improvement in runtime.

✉ Dean A. Bodenham
dean.bodenham@imperial.ac.uk

1 Department of Mathematics, Imperial College London, South Kensington Campus, London SW7 1AZ, UK

2 RIKEN Center for Advanced Intelligence Project, 1-4-1 Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan

3 Institute of Mathematics for Industry, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan

Table 1 Computational complexity MMD and euMMD, with $n = n_1 + n_2$ samples and L permutations.

Algorithm	Statistic	p-value
MMD	$\mathcal{O}(n^2)$	$\mathcal{O}(Ln^2)$
euMMD (proposed)	$\mathcal{O}(n \log n)$	$\mathcal{O}(n \log n + Ln)$

One concern may be that reliance on the Laplacian kernel is overly restrictive, and another kernel may be preferred; in Fig. 1 it is shown that for univariate, real-valued data the performance of the MMD two-sample test with the Laplacian kernel is broadly the same as when the popular Gaussian kernel is used.

A greater influence on performance is, in fact, the choice of value for the kernel parameter, as shown in Sect. S1 of the Supp. Material. A common method for setting the kernel parameter value is the so-called *median heuristic*; Sect. 3.1 discusses how for univariate data the kernel parameter can be set using this heuristic in $\mathcal{O}(n \log n)$ using existing algorithms from the literature.

Along with the MMD, another popular two-sample test with an extensive literature is the *energy distance* (Baringhaus and Franz 2004; Székely and Rizzo 2004, 2013). Its formulation is very similar to that of the MMD, and recent work has shown that the two tests are closely related (Sejdinovic et al. 2013; Shen and Vogelstein 2018). However, given the choice, one may prefer to use the MMD since there are cases where the MMD is more powerful than the energy distance (Sejdinovic et al. 2013); see Fig. 4 and Sect. 2 for a discussion.

There is in fact a class of "energy statistics" (Székely and Rizzo 2013), one of which is the *distance correlation* (Székely et al. 2007). Inspired by an efficient method for computing Kendall's τ coefficient (Knight 1966), recent work (Huo and Székely 2016) has shown that an efficient computation of the distance correlation can be found for univariate real-valued data after first sorting the data. That approach prompted this current work, in order to see if a similar approach could be used to find an efficient MMD two-sample test.

While the proposed method is exact for univariate data, it does not easily generalise to the d-dimensional multivariate case. However, one could use random projections (Cuesta-Albertos et al. 2006; Rahimi and Recht 2007; Wei et al. 2016) to project the data onto one dimension and obtain an average value for the MMD over multiple projections; this was done in Huang and Huo (2017) for the Energy Distance. Another approach is to apply a univariate test to the distances between the observations and fixed "centre points" in d-dimensional space (Heller and Heller 2016). These two different approaches to using a univariate test on multivariate data are explored in Sect. 4 using euMMD as the base univariate two-sample test, although several experiments are relegated to the appendix as this comparison is not the primary motivation for this work.

The rest of the paper is organized as follows: Sect. 2 provides definitions of the MMD statistic and the Laplacian kernel. Section 3 describes our proposed algorithm, euMMD, as well as related work and experiments showing the improved speed-up on synthetic data. Section 4 describes

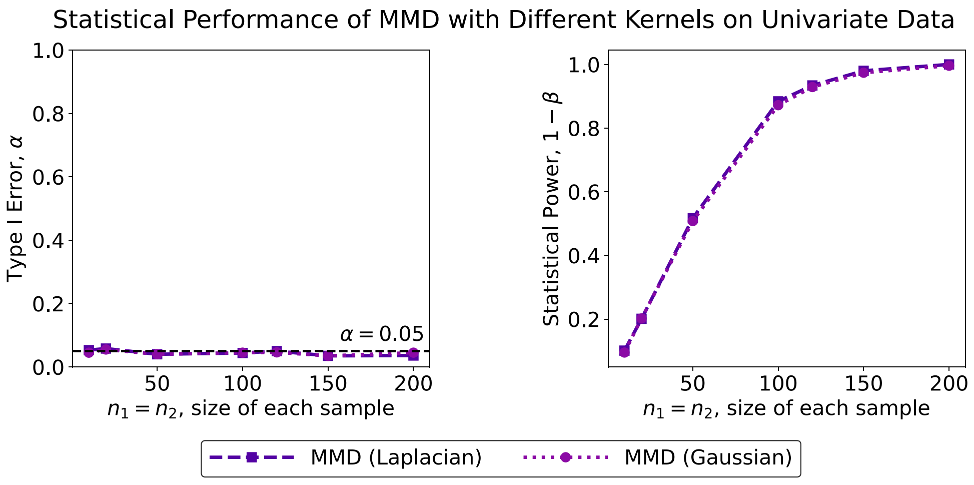


Fig. 1 A comparison of Type I error and statistical power for the (i) MMD with the Laplacian kernel and (ii) The MMD with the Gaussian kernel, for samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$. **(Left)** the Type I error is computed when the underlying distributions of $\mathbf{x}$ and $\mathbf{y}$ are both i.i.d. $\Gamma(1, 1)$. **(Right)** the statistical error is computed when $\mathbf{x}$ is i.i.d. $\Gamma(1, 1)$ and $\mathbf{y}$ is i.i.d. $\mathrm{N}(1, 1)$. We note that in this experiment the performance of the MMD approach is almost identical, whether the Laplacian or Gaussian kernel is used. Moreover, the MMD test has more power the energy distance test, at least for this data. In both cases, a significance threshold of $\alpha = 0.05$ is used with $L = 1000$ permutations to compute a p-value, and the performance shown is the average performance over 100 trials

the approximate extension to the d-dimensional case, and provides experiments showing the good performance of the approximate method. Several proofs and additional experiments are contained in the Supplementary Material. Note that any reference to computational complexity refers to *worst-case* computational complexity (see, e.g. Cormen et al. (2009)).

2 Background

Given a space $\mathcal{X}$ and two samples $\mathbf{x}, \mathbf{y} \subset \mathcal{X}$, with $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$, and a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the minimum variance unbiased estimate of the squared maximum mean discrepancy (MMD) statistic was defined in Borgwardt et al. (2006); Gretton et al. (2012a) as

$$\begin{aligned}\mathrm{MMD}^2(\mathbf{x}, \mathbf{y}) = &\frac{1}{n_1(n_1-1)} \sum_{i=1}^{n_1} \sum_{\substack{j=1 \\ j \neq i}}^{n_1} k(x_i, x_j) \\ &+ \frac{1}{n_2(n_2-1)} \sum_{p=1}^{n_2} \sum_{\substack{q=1 \\ q \neq p}}^{n_2} k(y_p, y_q) \\ &- \frac{2}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{p=1}^{n_2} k(x_i, y_p).\end{aligned} \tag{1}$$

Note that is more convenient to deal with $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$, the *squared* MMD statistic, rather than the actual (unsquared) MMD statistic, and in this paper we shall often refer to *the MMD statistic* when we really mean $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$.

The statistic in Eq. (1) is *unbiased* and is denoted by MMD_u^2 in Gretton et al (2012a, [Eq. (3)]) because there is a biased version MMD_b^2 which is very similar. Whichever version is used does not make a difference in terms of the computational complexity, and we shall focus on the unbiased version, simply denoting it by $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$ or MMD^2 when the underlying sets are understood. If one assumes that the kernel function is $\mathcal{O}(d)$, then it is clear that the computational complexity of computing MMD^2 directly from Eq. (1) is $\mathcal{O}(d(n_1^2 + n_2^2 + n_1 n_2))$, which is the same as $\mathcal{O}(dn^2)$, after defining $n = n_1 + n_2$.

For the MMD statistic to be most effective, it is important to choose a kernel that is *characteristic* on $\mathbb{R}^d$ (Sriperumbudur et al. 2010; Fukumizu et al. 2009). This is a property that means the kernel can be used to detect any change in a distribution (Fukumizu et al. 2004, 2008).

Two popular characteristic kernels are the Gaussian kernel and Laplacian kernel (Fukumizu et al. 2009; Sriperumbudur et al. 2010). For real d-dimensional vectors $z, z' \in \mathbb{R}^d$, with $z = \{z_1, z_2, \ldots, z_d\}$ and $z' = \{z'_1, z'_2, \ldots, z'_d\}$, for parameters $\beta, \gamma > 0$ the Laplacian kernel $k_{\mathrm{L},\beta}$ and the Gaussian kernel $k_{\mathrm{G},\gamma}$ are defined as

$$\begin{aligned}k_{\mathrm{L},\beta}(z, z') &= \exp\left(-\beta \|z - z'\|_1\right) \\ &= \exp\left(-\beta \sum_{\alpha=1}^{d} \left|z_\alpha - z'_\alpha\right|\right),\end{aligned} \tag{2}$$

$$\begin{aligned}k_{\mathrm{G},\gamma}(z, z') &= \exp\left(-\gamma \|z - z'\|_2^2\right) \\ &= \exp\left(-\gamma \sum_{\alpha=1}^{d} (z_\alpha - z'_\alpha)^2\right).\end{aligned} \tag{3}$$

In the univariate case with observations $z, z' \in \mathbb{R}$, the Laplacian kernel in Eq. (2) reduces to

$$k_{\mathrm{L},\beta}(z, z') = \exp(-\beta \left|z - z'\right|). \tag{4}$$

In Sect. 3 this kernel is used to define an exact $\mathcal{O}(n \log n)$ method for computing MMD^2 in Eq. (1) for univariate real-valued data. We call this algorithm `euMMD`. Note that in the remainder of this section and Sect. 3 the focus will be on univariate sets, while in Sect. 4 we look at approaches for extending this work to multivariate sets. In each case, we make explicit whether a value $z \in \mathbb{R}$ or $z \in \mathbb{R}^d$ for $d > 1$.

2.1 The null distribution of MMD^2

Under the null hypothesis that the data $\mathbf{x}$ and $\mathbf{y}$ are observations of i.i.d. random variables sampled from the same distribution, the statistic $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$ can be shown in the limit to follow the distribution of an infinite sum of weighted (non-central) chi-squared random variables, which can be approximated by moment-matching with Pearson curves Gretton et al. (2009, 2012a). However, computing the coefficients needed to use these approximations will be $\mathcal{O}(n^3)$, where as above $n = n_1 + n_2$, and $\mathbf{x}$ and $\mathbf{y}$ have n_1 and n_2 elements, respectively.

Computationally, and perhaps theoretically, it is preferable to use permutations to compute the empirical distribution of the statistic given the data, since this approach does not rely on approximations. One then obtains a p-value by ranking the statistic within the empirical distribution; see Sect. S4 in the Supp. Material for details. In Sect. 3 it is shown that computing $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$ statistic with the `euMMD` algorithm is $\mathcal{O}(n \log n)$, and so if L permutations are used, computing a p-value using permutations is $\mathcal{O}(Ln \log n)$.

2.2 Choice of kernel and other tests

The Gaussian kernel in Eq. (3) is often a popular choice when computing the MMD, and this preference may be due to the two-norm being invariant to rotations of multivariate data. However, we do not require such a property for univariate data, and then a preferred choice of kernel may be less clear.

In Fig. 1, the right panel shows that the performance of the MMD test with the Laplacian and Gaussian kernels is almost identical for detecting a change in distribution between a $N(1, 1)$ and $\Gamma(1, 1)$ distribution; note this is a case where the two distributions both have mean and variance equal to 1, but the shape of the distributions is different. The energy distance (Baringhaus and Franz 2004; Székely and Rizzo 2004, 2013) is also included in this comparison; the energy distance statistic has the same form as the MMD^2 in Eq. (1) with $k(z, z') = -\|z - z'\|_2$. The left panel of Fig. 1 shows the Type I error of the three tests when the underlying distributions for both $\mathbf{x}$ and $\mathbf{y}$ are a $\Gamma(1, 1)$ distribution. In both panels a significance threshold of $\alpha = 0.05$ is used with $L = 100$ permutations to compute the p-value, and each point on the plot represents the average Type I error/power over 1000 trials. The kernel parameter value is set using the median heuristic as described in Sect. 3.1.

This experiment provides two insights: (a) The performance of the MMD for univariate data is very similar whether the Laplacian or Gaussian kernel is used, and (b) The MMD has more power than the energy distance, at least in this case where the data $\mathbf{x}$ and $\mathbf{y}$ are sampled from the $N(1, 1)$ and $\Gamma(1, 1)$ distributions; Fig. S9 in the Supp. Material shows a similar result for normally-distributed data, except that the energy distance has slightly higher power than the MMD methods.

3 Proposed algorithm: `euMMD`

This section describes the proposed `euMMD` algorithm for efficiently computing MMD^2 for univariate data. We start with, for any $m \geq 1$,

Lemma 1 *Given a set* $\{z_1, z_2, \ldots, z_m\} \subset \mathbb{R}$ *and a* symmetric *function* $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, *then*

$$\sum_{i=1}^{m} \sum_{j=1}^{i} f(z_i, z_j) = \sum_{i=1}^{m} \sum_{j=i}^{m} f(z_i, z_j). \tag{5}$$

The proof follows immediately from the summation laws, and is included in Sect. S5.1 of the Supp. Material. However, Lemma 1 is a key step in later results. In particular, it leads directly to

Lemma 2 *Given a set* $\{z_1, z_2, \ldots, z_m\} \subset \mathbb{R}$ *and a symmetric function* $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, *then*

$$\sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} f(z_i, z_j) = 2 \sum_{i=1}^{m} \sum_{j=1}^{i-1} f(z_i, z_j). \tag{6}$$

Proof See Sect. S5.2 of the Supp. Material. □

Now, since every kernel function is symmetric (Shawe-Taylor and Cristianini 2004), Lemma 2 allows us to rewrite MMD^2 as follows:

$$\text{MMD}^2(\mathbf{x}, \mathbf{y}) = 2\left(\alpha_1 T_1 + \alpha_2 T_2 + \alpha_3 T_3\right), \tag{7}$$

where

$$T_1 = \sum_{i=1}^{n_1} \sum_{j=1}^{i-1} k(x_i, x_j), \qquad \alpha_1 = \frac{1}{n_1(n_1 - 1)},$$
$$T_2 = \sum_{p=1}^{n_2} \sum_{q=1}^{p-1} k(y_p, y_q), \qquad \alpha_2 = \frac{1}{n_2(n_2 - 1)},$$
$$T_3 = \sum_{i=1}^{n_1} \sum_{p=1}^{n_2} k(x_i, y_p), \qquad \alpha_3 = -\frac{1}{n_1 n_2}.$$

Remark 3 Essentially, we have simply shown that only the lower-triangular elements of the kernel matrix are needed to compute the MMD test statistic, but this observation will be key in deriving the new efficient method.

Using Eq. (7) for $\text{MMD}^2(\mathbf{x}, \mathbf{y})$, let us look at the lower triangular terms of the kernel matrix for the set $\{x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2}\}$, as shown in Fig. 2.

Notice that adding together the terms *below* the main diagonal in the top-left submatrix gives T_1. Similarly, adding the terms *below* the main diagonal in the bottom-right submatrix gives T_2. T_3 is obtained by adding all the terms in the bottom-left submatrix; however, if we define T_4 to be the sum of all the terms in the *entire* kernel matrix *below* the main diagonal, then

$$T_4 = T_1 + T_2 + T_3 \qquad \Rightarrow T_3 = T_4 - T_1 - T_2. \tag{8}$$

Therefore, if we could find an efficient method for computing the sum of lower-triangular terms in a kernel matrix, we would have an efficient method for computing MMD^2. One next realises that, when computing these lower-triangular sums, the order in which the terms are added does not matter. This observation is described in

Lemma 4 *Given a set* $\{z_1, z_2, \ldots, z_m\} \subset \mathbb{R}$, *a symmetric function* $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *and a permutation* $\sigma : \{1, 2, \ldots, m\} \to \{1, 2, \ldots, m\}$, *then:*

$$\sum_{i=1}^{m} \sum_{j=1}^{i-1} f(z_i, z_j) = \sum_{i=1}^{m} \sum_{j=1}^{i-1} f(z_{\sigma(i)}, z_{\sigma(j)}). \tag{9}$$

Proof See Sect. S5.3 of the Supp. Material. □

$$
\begin{array}{c|ccccc:cccc|c}
 & \multicolumn{5}{c}{\overbrace{\hphantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}^{n_1}} & & & & & \\
 & k(x_1,x_1) & & & & & & & & & \\
 & k(x_2,x_1) & k(x_2,x_2) & & & & & & & & \\
n_1 & k(x_3,x_1) & k(x_3,x_2) & k(x_3,x_3) & & & & & & & \\
 & \vdots & \vdots & \vdots & \ddots & & & & & & \\
 & k(x_{n_1},x_1) & k(x_{n_1},x_2) & k(x_{n_1},x_3) & \cdots & k(x_{n_1},x_{n_1}) & & & & & \\
\hdashline
 & k(y_1,x_1) & k(y_1,x_2) & k(y_1,x_3) & \cdots & k(y_1,x_{n_1}) & k(y_1,y_1) & & & & \\
 & k(y_2,x_1) & k(y_2,x_2) & k(y_2,x_3) & \cdots & k(y_2,x_{n_1}) & k(y_2,y_1) & k(y_2,y_2) & & & \\
n_2 & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \ddots & & n_2 \\
 & k(y_{n_2},x_1) & k(y_{n_2},x_2) & k(y_{n_2},x_3) & \cdots & k(y_{n_2},x_{n_1}) & k(y_{n_2},y_1) & k(y_{n_2},y_2) & \cdots & k(y_{n_2},y_{n_2}) & \\
 & \multicolumn{5}{c}{\underbrace{\hphantom{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}}_{n_1}} & \multicolumn{4}{c}{\underbrace{\hphantom{xxxxxxxxxxxxxxxxxxxx}}_{n_2}} &
\end{array}
$$

Fig. 2 The kernel matrix for the set $\{x_1, x_2, \ldots, x_{n_1}, y_1, y_2, \ldots, y_{n_2}\}$ and kernel k. Notice how T_1, T_2 and T_3 are obtained by adding the visible terms in the upper-left, lower-right and lower-left submatrices, respectively (but excluding the main diagonal terms for T_1 and T_2)

Since a permutation $\sigma : \{1, 2, \ldots, m\} \to \{1, 2, \ldots, m\}$ is simply a bijection on $\{1, 2, \ldots, m\}$, Lemma 4 holds in particular for the permutation that orders a specific set of elements $\{z_1, z_2, \ldots, z_m\}$ in increasing value, i.e. the permutation $\sigma(i) \to (i)$, where $z_{(1)} \le z_{(2)} \le \cdots \le z_{(m-1)} \le z_{(m)}$. This leads to the key result using the Laplacian kernel described in

Proposition 5 *Given a set of observations* $\{z_1, z_2, \ldots, z_m\} \subset \mathbb{R}$, *let* $z_{(i)}$ *denote the* i*th smallest element of the set, so that* $z_{(1)} \le z_{(2)} \le \cdots \le z_{(m)}$. *Then, using the Laplacian kernel with parameter* $\beta > 0$, *the quantity*

$$
S \underset{\text{def}}{=} \sum_{i=1}^{m} \sum_{j=1}^{i-1} k_{\mathrm{L},\beta}(z_i, z_j) = \sum_{i=1}^{m} \sum_{j=1}^{i-1} \exp\left(-\beta\left[z_{(i)} - z_{(j)}\right]\right),
$$

can be computed recursively by defining $S = S_m$ *and then using the equations* $S_1 = R_1 = 0$ *and, for* $k \in \{2, \ldots, m\}$,

$$
\begin{aligned}
D_k &= \exp\left(-\beta\left[z_{(k)} - z_{(k-1)}\right]\right), \\
R_k &= (R_{k-1} + 1) \cdot D_k, \\
S_k &= S_{k-1} + R_k.
\end{aligned}
\tag{10}
$$

Proof Using Lemma 4,

$$
\begin{aligned}
S &= \sum_{i=1}^{m} \sum_{j=1}^{i-1} k_{\mathrm{L},\beta}(z_{(i)}, z_{(j)}) \\
&= \sum_{i=1}^{m} \sum_{j=1}^{i-1} \exp(-\beta(z_{(i)} - z_{(j)})),
\end{aligned}
$$

since $z_{(i)} - z_{(j)} \ge 0$ for $j < i$. For $k \in \{1, 2, \ldots, m\}$, if we define D_k as in Eq. (10) and

$$
\begin{aligned}
S_k &= \sum_{i=1}^{k} \sum_{j=1}^{i-1} \exp(-\beta(z_{(i)} - z_{(j)})), \\
R_k &= \sum_{j=1}^{k-1} \exp(-\beta(z_{(k)} - z_{(j)})),
\end{aligned}
$$

one immediately obtains $S_1 = R_1 = 0$ and derives the recursive equations in Eq. (10). For the full details, see Sect. S5.4 of the Supp. Material. □

Proposition 5 leads directly to

Corollary 6 *Given a set of observations* $\{z_1, z_2, \ldots, z_m\} \subset \mathbb{R}$, *and a parameter* $\beta > 0$, *then*

$$
S = \sum_{i=1}^{m} \sum_{j=1}^{i-1} k_{\mathrm{L},\beta}(z_i, z_j)
$$

can be computed in $\mathcal{O}(m \log m)$ *time.*

Proof The observations are first sorted to $z_{(1)}, \ldots, z_{(m)}$, using a worst-case $\mathcal{O}(m \log m)$ sorting algorithm such as merge sort. The sequential update equations for S, given in Eq. (10), are all $\mathcal{O}(1)$, so together the $m-1$ recursive updates are $\mathcal{O}(m)$, and so overall the algorithm is $\mathcal{O}(m \log m)$. □

Remark 7 In plain terms, Corollary 6 shows that one can compute the sum of lower-triangular terms in a kernel matrix, for univariate data and while using the Laplacian kernel, in $\mathcal{O}(m \log m)$.

Algorithm 1 TriSSL: Triangular Sorted Sum of Laplacians

```
Input: $\beta > 0$; data $\mathbf{z} = [z_1, z_2, \ldots, z_n]$ with $z_i \leq z_{i+1}$
1: Initialise $D = R = S = 0$
2: for $i = 2$ to $n$ do
3:     $D = \exp\left(-\beta\left[z_i - z_{i-1}\right]\right)$
4:     $R = (R + 1) \cdot D$
5:     $S = S + R$
6: end for
Output: $S$
```

Algorithm 2 euMMD: Efficient computation of univariate MMD^2. Note that pseudocode for MergeTwoAlreadySorted can be found in Sect. S7 of the Supp. Material

```
Input: $\beta > 0$; data $\mathbf{x} = [x_1, x_2, \ldots, x_{n_1}]$ and data $\mathbf{y} = [y_1, y_2, \ldots, y_{n_2}]$.
1: $\mathbf{x}$ = MergeSort($\mathbf{x}$)
2: $\mathbf{y}$ = MergeSort($\mathbf{y}$)
3: $T_1$ = TriSSL($\mathbf{x}, \beta$)
4: $T_2$ = TriSSL($\mathbf{y}, \beta$)
5: $\mathbf{z}$ = MergeTwoAlreadySorted($\mathbf{x}, \mathbf{y}$)
6: $T_4$ = TriSSL($\mathbf{z}, \beta$)
7: $T_3 = T_4 - T_1 - T_2$
8: $M = T_1/(n_1(n_1 - 1)) + T_2/(n_2(n_2 - 1)) - T_3/(n_1 n_2)$
Output: $2 \cdot M$
```

Pseudocode implementing the recursive update equations given in Eq. (10) is given in the TriSSL subroutine in Algorithm 1. We can now state the main result:

Proposition 8 *Given two samples* $\mathbf{x}, \mathbf{y} \subset \mathbb{R}$ *containing a total of* n *observations, the* euMMD *algorithm computes the exact* MMD$^2(\mathbf{x}, \mathbf{y})$ *statistic defined in Eq. (1) using the Laplacian kernel in* $\mathcal{O}(n \log n)$ *time and* $\mathcal{O}(n)$ *space.*

Proof Let $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$ denote the two samples and define $n = n_1 + n_2$. First, the values in $\mathbf{x}$ are sorted to obtain the sorted set $\{x_{(1)}, x_{(2)}, \ldots, x_{(n_1)}\}$ and then the recursive formulae in Eq. (10) of Prop. 5 are applied to these sorted values to compute T_1. The same steps are followed for the values in $\mathbf{y}$ to compute T_2. This is summarised by lines 1-4 in the pseudocode in Algorithm 2, which describes the procedure in detail; the TriSSL subroutine, as described in Algorithm 1, implements the update equations in Eq. (10). (Note that the notation $\mathbf{x}$ is used in the pseudocode to denote an array containing the values $x_1, \ldots, x_{n_1}$, while here $\mathbf{x}$ denotes the set.) The sorted values are then merged into a sorted set $\mathbf{z}$ by the MergeTwoAlreadySorted algorithm in $\mathcal{O}(n)$ time (see Sect. S7 of the Supp. Material), and then TriSSL computes T_4 from $\mathbf{z}$. Equation (8) then gives $T_3 = T_4 - T_1 - T_2$. With T_1, T_2 and T_3 computed, Eq. (7) gives MMD$^2(\mathbf{x}, \mathbf{y})$. To analyse the complexity of euMMD, we look at the pseudocode in Algorithm 2. The merge sorts in lines 1-2 are $\mathcal{O}(n \log n)$ (Cormen et al. 2009).

Lines 3-6 are all $\mathcal{O}(n)$, and lines 7-8 are all $\mathcal{O}(1)$. Therefore, euMMD is $\mathcal{O}(n \log n)$ in time overall. Since merge sort and TriSSL both have space complexity $\mathcal{O}(n)$, euMMD also has space complexity $\mathcal{O}(n)$. □

Remark 9 Note that it is not necessary to use the MergeSort algorithm and the MergeTwoAlreadySorted subroutine; any other $\mathcal{O}(n \log n)$ sorting procedure could be used for $\mathbf{x}$ and $\mathbf{y}$, which could be concatenated to form $\mathbf{z}$, and then $\mathbf{z}$ could be sorted with the same algorithm. The computational complexity would stil be $\mathcal{O}(n \log n)$. However, the $\mathcal{O}(n)$ MergeTwoAlreadySorted step takes advantage of $\mathbf{x}$ and $\mathbf{y}$ being already sorted.

Remark 10 When analysing the euMMD algorithm one notices that, after the data is sorted, the remaining part of the algorithm is $\mathcal{O}(n)$. In other words, the only $\mathcal{O}(n \log n)$ step is the initial sorting of the data, where a sorting algorithm with worst-case time complexity $\mathcal{O}(n \log n)$ (also denoted $\Omega(n \log n)$), such as merge sort (Knuth 1997b; Cormen et al. 2009), is used; it is well-known that $\Omega(n \log n)$ is the lower bound on the time complexity for sorting general real-valued data (Cormen et al. 2009). However, one then realises that in special cases where the two samples are already sorted, for example in cases where the data are ordered times, the euMMD algorithm is then linear in n. Another special case is when the data belong to some restricted set, e.g. the data are integers in a known finite range, such as for a multinomial distribution with a known number of categories (for which the MMD two-sample test would be valid), then a linear time sorting method such as counting sort (Cormen et al. 2009) could be used, which would give the algorithm linear-time computational complexity. We emphasise, however, that these are special cases and do not focus on them here.

Remark 11 Note that there is an efficient method for computing the Energy Distance (Huang and Huo 2017), related to the approach in Huo and Székely (2016), which is very similar to the algorithm for the proposed efficient MMD presented here. However, there are at least a few key differences: our approach to the computation of the mixed T_3 term is much simpler, our sequential update approach utilizing the TriSSL algorithm is much clearer, and this sequential approach allows the use of the Laplacian kernel, which would be nontrivial to employ in the algorithm in Huang and Huo (2017); furthermore, a careful proof showing the correctness of euMMD is provided.

3.1 Setting the kernel parameter

Most kernels require a user-defined parameter to be selected in advance of the computation. For the Laplacian kernel, this is a parameter $\beta > 0$. While it remains an open question how to select the optimal value of β (or indeed the value of other kernels' parameters), a commonly-used approach is to use the *median heuristic*, where for a set of values $\{z_1, z_2, \ldots, z_n\}$, one computes

$$h = \text{median}\{|z_i - z_j| : i, j \in \{1, 2, \ldots, n, i < j\}\}$$

and then one sets $\beta = \frac{1}{h}$. A naive computation of h would require the computation of all $\frac{n}{2}(n-1)$ differences, resulting in a computational complexity of $\mathcal{O}(n^2)$. However, a similar quantity has been previously considered as a measure of dispersion (Shamos 1976, Theorem 3.6), (Bickel and Lehmann 1979, Example 9) (Rousseeuw and Croux 1993), and there exists an efficient method for computing h in $\mathcal{O}(n \log n)$ (Croux and Rousseeuw 1992, Sect. 3), based on Johnson and Mizoguchi (1978). Note that although this method is worst-case $\mathcal{O}(n \log n)$, it is based on the repeated use of the selection algorithm for finding the kth smallest value among n values in worst-case $\mathcal{O}(n)$ (Cormen et al. 2009, Sect. II.9.3). Needing to compute the median heuristic value first makes the algorithm slightly more computationally expensive overall compared to only computing the MMD^2 statistic, despite both algorithms being $\mathcal{O}(n \log n)$, because of larger leading coefficients hidden by the big-O notation. However, when used in combination with the computation of the p-value via permutations, this difference ends up being very slight; see Fig. S12 in the Supp. Material.

While it has been shown that the median heuristic may not necessarily provide the "best" choice for the parameter (Ramdas et al. 2015), i.e. the parameter value that leads to the greatest power for the MMD test, it generally works well; see Sect. S1 in the Supp. Material for additional experiments. There are other schemes for selecting β (Sriperumbudur et al. 2009; Gretton et al. 2012b), but these amount to testing various values and selecting the one which "works best" for the given data. We do not pursue the question of how to set the kernel parameter further here.

3.2 Computing *p*-values

The previous sections have shown that the MMD statistic for univariate data can be computed in $\mathcal{O}(n \log n)$, along with the median heuristic for setting the kernel parameter in $\mathcal{O}(n \log n)$. In Sect. 2.1 it is discussed how it may be preferable to use L permutations to compute a p-value for the MMD test statistic, which would result in an algorithm that is $\mathcal{O}(Ln \log n)$. This approach involves first computing the median heuristic to determine the kernel parameter β and then computing the test statistic MMD$^2(\mathbf{x}, \mathbf{y})$. Then, one concatenates $\mathbf{x}$ and $\mathbf{y}$ to form the vector $\mathbf{z}$, randomly permutes the values in $\mathbf{z}$ to create vector $\widetilde{\mathbf{z}}$, before subsetting this vector to $\widetilde{\mathbf{x}}$ (first n_1 values) and $\widetilde{\mathbf{y}}$ (last n_2 values). One calls euMMD to compute MMD$^2(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}})$. Repeating this process for a total of L permutations, one counts the number of MMD$^2(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}})$ values smaller than MMD$^2(\mathbf{x}, \mathbf{y})$ as c, and then computes

Table 2 Runtime comparison of MMD and euMMD univariate data with $n_1 = n_2 = 2000$ in Python

Algorithm	Time (seconds)	Relative speedup to MMD
MMD	32.341	$1\times$
euMMD (proposed)	0.034	$\sim 1000\times$

$p = \max\{1 - |1 - 2c/(L+1)|, p_{\min}\}$, where $p_{\min}$ is set as $1/(2(L+1))$ to avoid p-values of 0 (this would give issues when using Fisher's method to combine p-values in the multivariate approximate method). It is apparent that this method is $\mathcal{O}(Ln \log n)$. Although the median heuristic only needs to be computed once, as the kernel parameter is the same for all permutations, this method requires sorting each pair of permuted samples $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{y}}$.

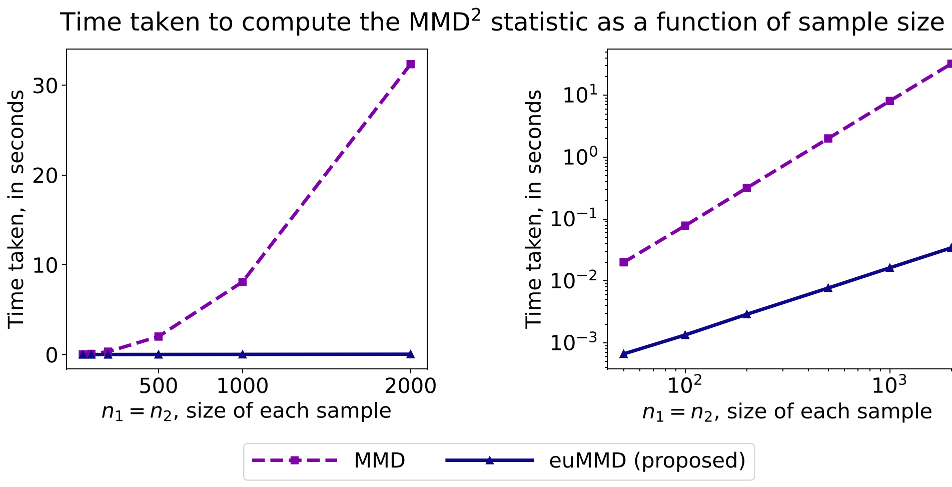
However, we can improve on this approach. Motivated by the observation that the euMMD algorithm is $\mathcal{O}(n)$ if the data is already sorted, we devise a simple approach to form permuted samples such that $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{y}}$ are already sorted. First, the kernel parameter (via the median heuristic) and MMD$^2(\mathbf{x}, \mathbf{y})$ are computed with euMMD in an $\mathcal{O}(n \log n)$ step. A result of this computation is the vector $\mathbf{z}$, which is a sorted vector containing the n values from $\mathbf{x}$ and $\mathbf{y}$. Instead of permuting the values in $\mathbf{z}$, an indicator vector $\boldsymbol{\iota} = (1, \ldots, 1, 0, \ldots, 0)$ is formed, where the first n_1 values are 1 and the remaining n_2 values are 0. For each permutation, the indicator vector $\boldsymbol{\iota}$ is permuted to $\widetilde{\boldsymbol{\iota}}$ rather than permuting the data in the vector $\mathbf{z}$; i.e. $\mathbf{z}$ remains sorted. Defining the function s such that $s(j, \widetilde{\boldsymbol{\iota}}) = k$ means the jth 1 in $\widetilde{\boldsymbol{\iota}}$ is at index k, the vector $\widetilde{\mathbf{x}}$ is defined as $\widetilde{\mathbf{x}} = \left(z_{s(1,\widetilde{\boldsymbol{\iota}})}, \ldots, z_{s(n_1,\widetilde{\boldsymbol{\iota}})}\right)$. The vector $\widetilde{\mathbf{y}}$ is similarly defined using s' where $s'(j, \widetilde{\boldsymbol{\iota}}) = k$ means the jth 0 in $\widetilde{\boldsymbol{\iota}}$ is at index k. Since $\mathbf{z}$ is sorted and s and s' are increasing functions in their first argument, the vectors $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{y}}$ are both sorted. Therefore, computing each MMD$^2(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}})$ will be $\mathcal{O}(n)$ and the overall complexity of computing the p-value after L permutations is $\mathcal{O}(n \log n + Ln)$. The pseudocode for this procedure is shown in Algorithm S2 in Sect. S6 in the Supp. Material (Table 2).

3.3 Runtime comparison

As discussed in Sect. 2, for $n = n_1 + n_2$, computing MMD^2 directly from Eq. (1) is $\mathcal{O}(n^2)$ for univariate data. On the other hand, Sect. 3 shows that MMD^2 can be computed in $\mathcal{O}(n \log n)$. Note that since euMMD is exact for univariate real-valued data, there is no need to compare its statistical performance to that of the naive computation of MMD^2, denoted MMD; see for example Gretton et al. (2012a) for a study of the statistical performance of MMD.

Figure 3 shows the improved performance of euMMD over MMD as n increases; the right-hand panel displays the same data as in the left-hand panel, but on a log-log scale. This

Fig. 3 A runtime comparison of Python implementations of the standard $\mathcal{O}(n^2)$ `MMD` and the proposed $\mathcal{O}(n \log n)$ `euMMD` on univariate data, as the size of the samples, $n_1 = n_2$, increases and $n = n_1 + n_2$. **Left:** linear scale. **Right** log-log scale. Table 1 shows the relative speedup for $n_1 = n_2 = 2000$



makes it easier to see the orders of magnitude difference between the two methods. Note that these are the average speeds over 10 trials, and error bars of one standard deviation are plotted, but are too small to see (i.e. little variation in time between trials).

Furthermore, Table 1 compares data in Fig. 3 when $n = 2000$, and shows that in this case `euMMD` is $\sim 1000\times$ faster than `MMD`. Note also that, in order to provide a fair comparison of the algorithms, we implemented the merge sort algorithm from Panny and Prodinger (1995) in Python.

Section S6 in the Supp. Material provides additional runtime experiments. One experiment shows that using the built-in Numpy sorting function would yield an extra $\sim 4\times$ speedup over using the "hand-coded" merge sort; however, this is probably an unfair comparison, since the Numpy sort function is implemented in C++. There are also runtime experiments for implementations in C++, which show more modest improvements in speed, up to about $124\times$.

While the results in Fig. 3 are illustrative of the improvement in computational complexity of `euMMD` over the standard `MMD`, another experiment showing a more practical implementation (a) Using C++, (b) Using the median heuristic with the kernel parameter for the Laplacian kernel, and (c) Using either 100 or 1000 permutations to compute the p-value, which show more modest improvements in speed of between $25\times$ to $328\times$; see Sect. S6.2 in the Supp. Material.

3.4 Statistical performance of MMD on univariate data

We conduct a number of experiments to evaluate the statistical performance of MMD on univariate data, and compare to other univariate two-sample testing methods: the Kolmogorov-Smirnov (Kolmogorov 1933; Smirnov 1948), Cramér-von Mises (Cramér 1928; von Mises 1928), Wilcoxon-Mann–Whitney (Wilcoxon 1945; Mann and Whitney 1947) energy distance (Baringhaus and Franz 2004; Székely and Rizzo 2004, 2013) and Student's t-test (Student, 1908). Figure 4 shows the results when there is a distribution change, the data in the first sample $\mathbf{x}$ are independent observations of a $\Gamma(1, 1)$ distribution, while the second sample $\mathbf{y}$ are independent observations of a $\mathrm{N}(1, 1)$ distribution (both samples have mean and variance equal to 1), the same as for Fig. 1. Figure 4 shows that the MMD has, in this case, higher power than the other methods.

Section S11 contains additional experiments, for another type of distribution shift, as well as a variance shift and mean shifts. The MMD performs the best for distribution shifts and joint-best for variance shifts, but does not perform as well as the other methods for mean shifts. Section S11 also contains figures showing the Type 1 error with y-axis on the scale $[0, 0.1]$.

4 Extension to multivariate data: `MEA-MMD`

While `euMMD` is an *exact* method for computing the MMD^2 statistic for univariate data, this approach does not easily extend to an exact method in the multivariate setting. However, in this section we extend our approach to an *approximate* methods for computing the MMD^2 statistic for multivariate data named `MEA-MMD`: *M*ultivariate *E*fficient *A*pproximate *MMD*.

We explore two approaches from the literature for adapting a univariate test to multivariate data: (a) using random projections as in Huang and Huo (2017), and (b) computing distances to a reference point as in Heller and Heller (2016). Both approaches involve projecting the observations from $\mathbb{R}^d \to \mathbb{R}$ and then performing the univariate test. For the remainder of this section, let $d \in \{2, 3, \ldots\}$ be the dimension and suppose we have samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\} \subset \mathbb{R}^d$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\} \subset \mathbb{R}^d$.

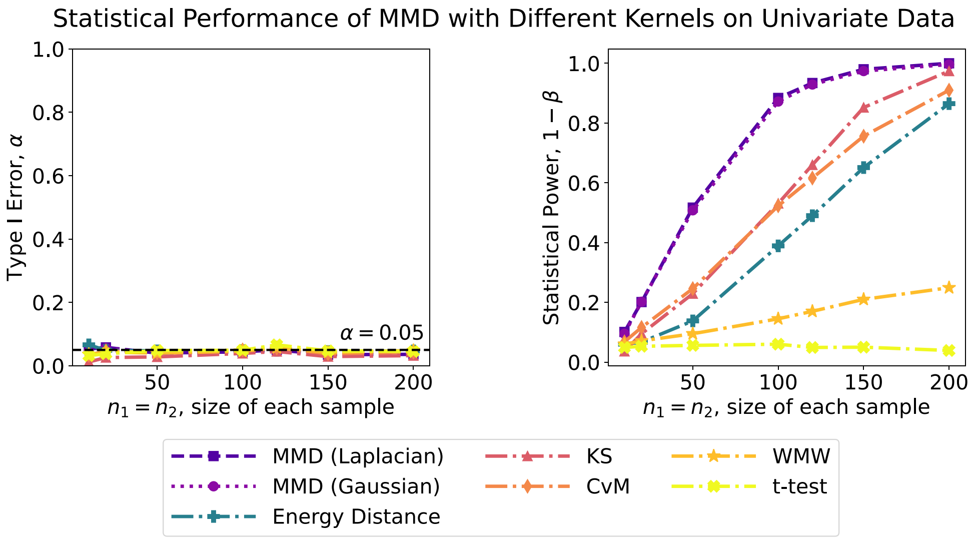


Fig. 4 A comparison of Type I error and statistical power for the (i) MMD with the Laplacian kernel and (ii) The MMD with the Gaussian kernel, (iii) The energy distance, (iv) The Kolmogorov-Smirnov test (KS), (v) The Cramér-von Mises test (CvM), (vi) The Wilcoxon-Mann–Whitney test (WMW), and (vii) Student's t-test, for samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$. **(Left)** the Type I error is computed when the underlying distributions of $\mathbf{x}$ and $\mathbf{y}$ are both i.i.d. $\Gamma(1, 1)$. **(Right)** the statistical error is computed when $\mathbf{x}$ is i.i.d. $\Gamma(1, 1)$ and $\mathbf{y}$ is i.i.d. $N(1, 1)$. We note that in this experiment the performance of the MMD approach is almost identical, whether the Laplacian or Gaussian kernel is used.In both cases, a significance threshold of $\alpha = 0.05$ is used with $L = 1000$ permutations to compute a p-value, and the performance shown is the average performance over 100 trials

4.1 Using random projections

Let $\mathcal{S}^{d-1}$ denote the unit sphere in $\mathbb{R}^d$, i.e.

$$\mathcal{S}^{d-1} = \left\{u \in \mathbb{R}^d \ : \ \|u\|_2 = 1\right\}$$

Following the approach in Huang and Huo (2017), instead of computing $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$, we can compute $\mathrm{MMD}^2(u_k^T\mathbf{x}, u_k^T\mathbf{y})$, for any $u_k \in \mathcal{S}^{d-1}$, where $u_k^T\mathbf{x} = \{u_k^T x_1, u_k^T x_2, \ldots, u_k^T x_{n_1}\} \subset \mathbb{R}$ and $u_k^T\mathbf{y} = \{u_k^T y_1, u_k^T y_2, \ldots, u_k^T y_{n_2}\} \subset \mathbb{R}$ are now univariate sets, using the euMMD algorithm. We can do this for K projections $u_k \in \mathcal{S}^{d-1}$, and then construct our statistic by averaging across the K projections, as in

$$\overline{\mathrm{MMD}_K^2}(\mathbf{x}, \mathbf{y}) = \frac{1}{K}\sum_{k=1}^{K}\mathrm{MMD}^2(u_k^T\mathbf{x}, u_k^T\mathbf{y}). \tag{11}$$

We call this algorithm MEA-MMD-Proj. The theoretical results in Huang and Huo (2017) for the energy distance can be adapted to demonstrate that $\overline{\mathrm{MMD}_K^2}(\mathbf{x}, \mathbf{y})$ converges to $\mathrm{MMD}^2(\mathbf{x}, \mathbf{y})$ by increasing K and the sample size.

4.2 Computing distances to a reference point

In Heller and Heller (2016), the multivariate vectors are projected onto univariate vectors by computing distances between the different individual observations. For $i \in \{1, 2, \ldots, n_1\}$, one computes the distances to x_i as follows:

$$\mathbf{x}^{(i)} = [\|z_1 - x_i\|_2, \ldots, \|x_{i-1} - x_i\|_2, \|x_{i+1} - x_i\|_2, \ldots, \|x_{n_1} - x_i\|_2]$$
$$\mathbf{y}^{(i)} = [\|y_1 - x_i\|_2, \|y_1 - x_i\|_2 \ldots, \|y_{n_2} - x_i\|_2].$$

Note that $\mathbf{x}^{(i)}, \mathbf{y}^{(i)} \subset \mathbb{R}$, with $n_1 - 1$ and n_2 observations, respectively; the ith component of $\mathbf{x}$ is removed rather than included as zero. Similarly, for $n_1 < i \leq n$, one can compute the distances between the observations and y_{i-n_1} to obtain $\mathbf{x}^{(i)}, \mathbf{y}^{(i)} \subset \mathbb{R}$ (where now $\mathbf{x}^{(i)}$ has n_1 observations and $\mathbf{y}^{(i)}$ has $n_2 - 1$ observations). Overall, there are $n_1 + n_2 = n$ projections. In Heller and Heller (2016) this was used to apply the univariate Kolmogorov-Smirnov and Anderson-Darling tests to multivariate data; here this approach will be used with the univariate MMD tests.

For notational convenience, we now assume there is a function $g : \mathbb{R} \to [0, 1]$ that computes a p-value given the value of a statistic $\mathrm{MMD}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$, although in practice this will be done via permutations of the $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$. We also use $h : [0, 1]^n \to [0, 1]$ to denote the p-value combination method due to Hommel (1983), which is briefly described in Sect. S9.4 of the Supp. Material. Then, after computing $p_i = g(\mathrm{MMD}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}))$ for $i = 1, 2, \ldots, n$, the final p-value is $p = h(p_1, p_2, \ldots, p_n)$. This method will be denoted MEA-MMD-Dist.

Of course, MEA-MMD-Dist and MEA-MMD-Proj are only approximations to $\text{MMD}^2(\mathbf{x}, \mathbf{y})$, so in addition to its runtime we need to evaluate the statistical performance of using the statistic in a statistical test, which we do in the next sections.

4.3 Statistical performance

We present simulations to investigate the Type I error and statistical power. In particular, we consider the case where the two samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$, where $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$ and $y_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,d})$, each $x_{i,j}$ is an observation of $X_{i,j}$, and $X_{i,j}$ are each i.i.d. normal; i.e. $X_{i,j} \sim \text{N}(\mu_1, \sigma_1^2)$. Similarly, each $y_{i,j}$ is an observation of $Y_{i,j}$, and the $Y_{i,j} \sim \text{N}(\mu_2, \sigma_2^2)$ are i.i.d. For the case of computing the Type I error, we set $\mu_1 = \mu_2 = 0$ and $\sigma_1 = \sigma_2 = 1$. For the case of computing the power, $\mu_1 = 0$, and $\sigma_1 = \sigma_2 = 1$, and $\mu_2 = \mu_1 + 0.5\sigma_1 = 0.5$. Note that the number of permutations used is $L = 100$, the threshold used is $\alpha = 0.05$, and the values are obtained over 100 independent trials. The results of this experiment are displayed in Fig. 5. Note that we have set the number of projections to be $K = 20$, in which case MEA-MMD-Proj should be computationally more efficient than MMD, as shown in Fig. 7. Sect. S8.1 in the Supp. Material contains a similar figure to Fig. 5, but with $K = 100$. Section S8.5 in the Supp. Material looks at variations of MEA-MMD-Dist approach. We notice from Fig. 5 that MEA-MMD-Proj has *slightly greater* power than MMD for this setting, at least for $n \leq 50$, while both have similar Type I errors, close to the threshold α. We have included MMD with both the Laplacian and Gaussian kernels, which have very similar performance.

4.3.1 Statistical performance: $d > n$

The experiment shown in Fig. 5 indicates one result of particular interest: the power of MEA-MMD-Proj is higher than the power of MMD for small $n_1 = n_2$.

We investigate this further by restricting $n_1, n_2 \in \{5, 10, 20, 30, 40\}$ with $n_1 = n_2$ and increase the dimension to $d = 100$ to consider the case $d > n$, where $n = n_1 + n_2$.

Again, $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\}$ are observations sampled as before, but for the power experiment, we set $\mu_2 = \frac{1}{\sqrt{40}}$ in order to keep the Kullback–Leibler divergence between the two distributions the same for this case where $d = 100$ as in the case where $d = 10$ in Fig. 5; see Sect. S8.2 in the Supp. Material for details.

Figure 6 shows that MEA-MMD-Proj has power $1 - \beta > 0.8$ for $d = 100$ and $n_1 = n_2 = 10$ and power close to 1 for $d = 100$ and $n_1 = n_2 = 20$, while the power for MMD is much lower in those cases, being around 0.2 and 0.4, respectively. This provides some evidence that MEA-MMD-Proj may have utility in the case $d > n$.

The poor performance of MEA-MMD-Dist may be due, as noted before, to the fact that the Hommel p-value combination procedure is conservative; see Sect. S8.5 in the Supp. Material for further details.

4.3.2 Comparison to other multivariate two-sample tests

We compare the performance of the proposed MEA-MMD-Proj to the following multivariate two-sample tests: the Cross-match test (Rosenbaum 2005), the Multivariate Runs test (Friedman and Rafsky 1979; Henze and Penrose 1999), and Hotelling's multivariate T-test (Hotelling 1931). We do not consider MEA-MMD-Dist any longer, since it has worse performance compared to MEA-MMD-Proj. We use the MMD with the Laplacian and Gaussian kernels and the energy distance as benchmarks. We discuss the implementations of the multivariate methods in Sect. S12 of the appendix.

Figure 6 compares these multivariate methods for a mean shift, with the number of dimensions $d = 100$. In this case, MEA-MMD-Proj has higher power than the other methods. The Multivariate Runs test has power equal to 0, possibly because $n_1 + n_2 < d$. The Hotelling test also power equal to 0, because $n_1 + n_2 < d$ and the sample covariance matrix is singular. In Sect. S12 of the Supp. Material, the same experiment is run for $d = 10$, and MEA-MMD-Proj again has superior power over these methods for small sample sizes.

Further experiments for a change in the mean shows that MEA-MMD-Proj performs well in comparison to the mutlivariate methods. For a change in variance or distribution, though, MEA-MMD-Proj does not perform well, and it is shown that only MMD performs well. Following a suggestion from one of the referees, we have applied the same random projections approach to other univariate tests used in Sect. 3.4. These random projection methods perform similarly to MEA-MMD-Proj, performing well for a change in the mean, but not for a change in variance or distribution. These experiments are also in Sect. S12 of the Supp. Material.

In summary, MEA-MMD-Proj may have utility for detecting a change in the mean, particularly when the number of dimensions is greater than the number of samples, but the multivariate MMD should be preferred for other cases.

4.4 Runtime comparison

Analysing the runtime of MEA-MMD-Proj by looking at the computation of Eq. (11), since computing $u_j^T\mathbf{x}$ and $u_j^T\mathbf{y}$ is $\mathcal{O}(dn)$, the overall computational complexity is $\mathcal{O}(Kn\log n + Kdn)$.

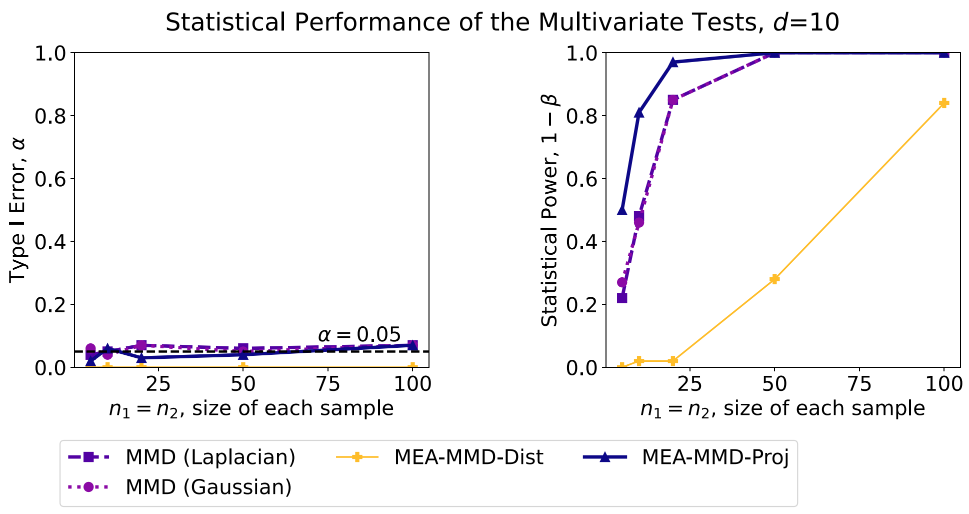


Fig. 5 A comparison of Type I error and statistical power for the (i) MMD with the Laplacian kernel, (ii) The MMD with the Gaussian kernel, (iii) The proposed `MEA-MMD-Proj` and (iv) The proposed `MEA-MMD-Dist`, for multivariate samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\} \subset \mathbb{R}^d$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\} \subset \mathbb{R}^d$ for $d = 10$. **(Left)** the Type I error is computed when the underlying distributions of each component of the x_i and y_j vectors are i.i.d. N(0, 1). **(Right)** the statistical error is computed when each component of the x_i are i.i.d. N(0, 1) and each component of the y_j are i.i.d. N(0.5, 1). In both cases, a significance threshold of $\alpha = 0.05$ is used with $L = 100$ permutations to compute a p-value, the number of projections for `MEA-MMD-Proj` is $K = 20$, and the performance shown is the average performance over 100 trials

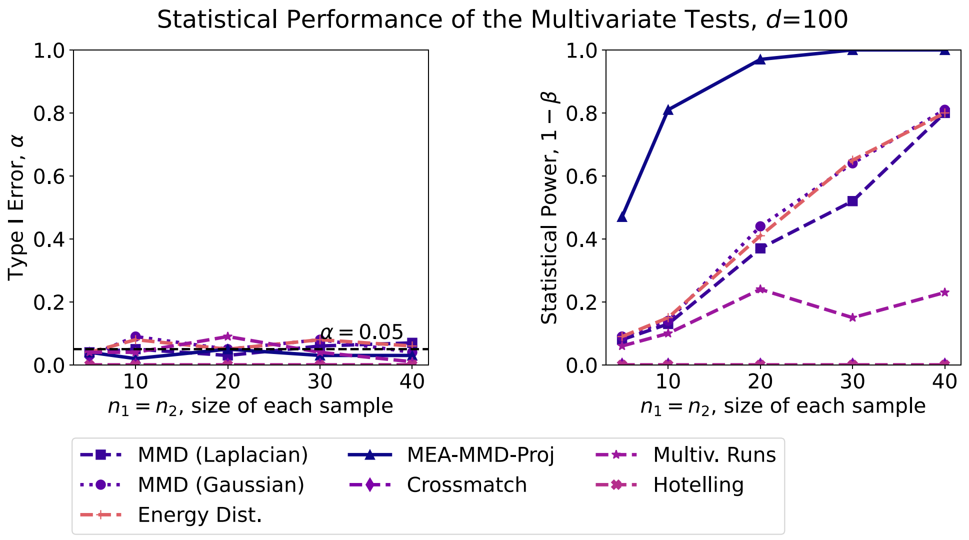


Fig. 6 A comparison of Type I error and statistical power for the (i) The proposed `MEA-MMD-Proj` and (ii) MMD with the Laplacian kernel, (iii) The MMD with the Gaussian kernel, (iv) The Crossmatch test, (v) The Multivariate Runs test, (vi) The energy distance, (vii) Hotelling's multivariate T-test, for multivariate samples $\mathbf{x} = \{x_1, x_2, \ldots, x_{n_1}\} \subset \mathbb{R}^d$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_{n_2}\} \subset \mathbb{R}^d$ for $d = 100$ and $n_1, n_2 \in \{10, 20, 30, 40\}$, and $n_1 = n_2$, so $d > n = n_1 + n_2$. **(Left)** the Type I error is computed when the underlying distributions of each component of the x_i and y_j vectors are i.i.d. N(0, 1). **(Right)** the statistical error is computed when each component of the x_i are i.i.d. N(0, 1) and each component of the y_j are i.i.d. N(μ_2, 1), where $\mu_2 = 1/\sqrt{40}$. In both cases, a significance threshold of $\alpha = 0.05$ is used with $L = 100$ permutations to compute a p-value, the number of projections for `MEA-MMD-Proj` is $K = 20$, and the performance shown is the average performance over 100 trials

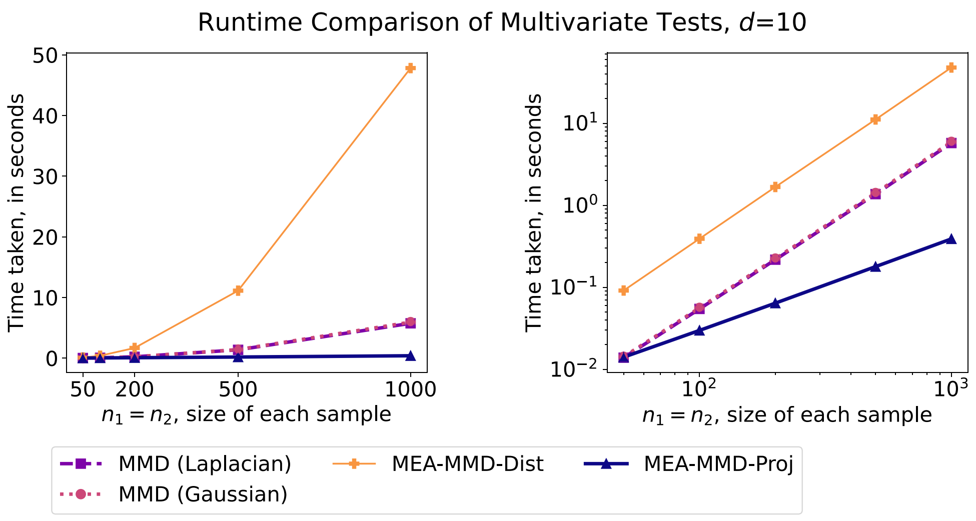


Fig. 7 Runtime comparison of C++ implementations of the multivariate (i) MMD with the Laplacian kernel, (ii) MMD with the Gaussian kernel, (iii) The proposed `MEA-MMD-Proj` and (iv) The proposed `MEA-MMD-Dist`, as $n_1 = n_2$ varies, with $d = 10$ and number of projections for `MEA-MMD-Proj` is $K = 20$. (**Left**) linear scale. (**Right**) log-log scale. Note that the p-values are computed using $L = 100$ permutations, but without using the median heuristic method to set the value of β

For `MEA-MMD-Dist`, since there are n computations of $\text{MMD}^2(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ (which are $\mathcal{O}(n \log n + dn)$ steps), and a single $\mathcal{O}(n \log n)$ for the Hommel p-value combination procedure, the whole algorithm is $\mathcal{O}(n^2 \log n + dn^2)$. This might seem counterproductive; after all, the standard MMD method is $\mathcal{O}(dn^2)$. However, as shown in Figs. 5 and 6 in Sect. 4.3, there are cases where these approximate approaches can have *greater* statistical power than the standard MMD.

Note that we are neglecting the number of permutations from the computational complexities, since we are assuming all methods will use the permutation approach. However, to be precise, `MEA-MMD-Dist` is $\mathcal{O}(Ldn^2 \log n)$, where L is the number of permutations.

Figure 7 investigates the runtime of `MEA-MMD-Proj` compared to the other methods by varying n and the number of projections K, and shows that when K is not too large, `MEA-MMD-Proj` can be significantly faster than `MMD`. In Fig. 7 we have used $K = 20$, as that seems to be sufficient for good performance; see Sect. S8.4 in the Supp. Material for details. Note also that Fig. 7 shows the runtime for when the p-values are computed, but without using the median heuristic; Fig. S14 in the Supp. Material shows that when the median heuristic is used, `MEA-MMD-Proj` is slightly slower than `MMD` for $n \leq 2000$. This is because the median heuristic algorithm needs to be computed for *every projection*. However, as shown in Sects. 4.3 and 4.3.1, we see that the utility of `MEA-MMD-Proj` is not its increased efficiency, but rather an increase of power in cases where n is small, and even when $n < d$.

4.4.1 Increasing the dimension d or the number of projections K

All the algorithms have computational complexity that is linear in the dimension of the data, d.

The computational complexity is clearly linear in K, but additional experiments in Sect. S8.4 in the Supp. Material show that for $K \geq 20$, the statistical performance of `MEA-MMD` is relatively constant, i.e. does not significantly improve for larger K.

4.5 Other approximate methods

Since `MMD` (Gretton et al. 2012a) has computational complexity $\mathcal{O}(dn^2)$, several approximate methods have been proposed which are more efficient.

In Gretton et al. (2012a) a linear-time approximate method `MMD-Linear` was proposed with computational complexity is $\mathcal{O}(dn)$, but this method has a higher variance than `MMD`, and so worse statistical performance. The `MMD-Block` method, known as B-test in Zaremba et al. (2013), approximates the MMD statistic by partitioning the kernel matrices into blocks; the size of the blocks, B, is a parameter for the method and we use their suggested heuristic $B = \sqrt{n_1}$, when $n_1 = n_2$ (as is the case in our simulation setup), and in this case its computational complexity is $\mathcal{O}(dn^{1.5})$. The `MMD-Fourier` method, known as *FastMMD* in Zhao and Meng (2015), uses the random Fourier features approach from Rahimi and Recht (2007); it has a parameter L, with a suggested heuristic for its value based on the sample size n, and it has computational complexity $\mathcal{O}(dnL)$.

However, while these methods are computationally more efficient, their statistical performance is worse than that of the original `MMD`, and so they are not considered in our comparison of statistical performance.

5 Discussion

While a direct computation of the MMD statistic for n univariate, real-valued samples is $\mathcal{O}(n^2)$, we present the algorithm euMMD, which can compute the *exact* MMD statistic with computational complexity $\mathcal{O}(n \log n)$ and storage complexity $\mathcal{O}(n)$, where these are *worst-case* complexities. This approach can lead to runtime improvements of several orders of magnitude.

Additionally, two *approximate* methods MEA-MMD-Proj and MEA-MMD-Dist, based on euMMD, are explored for computing the MMD statistic in the multivariate setting for d-dimensional data. The MEA-MMD-Proj approach depends on a user-defined number of random projections K, and for $K = 20$ this algorithm will have improved computational complexity and runtime performance for large n. However, of greater interest are the cases with small n, and even cases where $n < d$, where MEA-MMD-Proj has better statistical perfomance than the exact MMD method.

So, while the univariate case is exact and has improved computational complexity, the multivariate approximate method MEA-MMD-Proj has greater statistical power in cases where n is small, and even works well in cases where $n < d$.

One interesting question is whether the $\mathcal{O}(n \log n)$ computational complexity of euMMD can be improved for univariate real-valued data, i.e. is the computational complexity $\Omega(n \log n)$? In Remark 10 it is mentioned that if the data is known to be integer-valued, then an existing $\mathcal{O}(n)$ sort allows the algorithm to be $\mathcal{O}(n)$ overall; however, this is a special case.

5.1 Code and figures

The euMMD and MEA-MMD-Proj algorithms are available in the R package named eummd on CRAN and in the Python package named eummd on PyPI. Both packages share the same underlying core code written in C++.

Section S12 in the Supp. Material discusses the implementation of the multivariate methods.

All experiments were run on an Apple Macbook Air with an M1 processor and 16 GB of RAM.

Plots were produced in Python using the matplotlib (Hunter 2007) library.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11222-023-10271-x.

Acknowledgements We are grateful to the two referees for their comments and suggestions which have led to improvements in the paper.

References

Baringhaus, L., Franz, C.: On a new multivariate two-sample test. J. Multivar. Anal. **88**(1), 190–206 (2004)

Bickel, P.J., Lehmann, E.L.: Descriptive statistics for nonparametric models iv. spread. In: Jaroslav Hájek Memorial Volume. Springer, pp 519–526, (1979)

Borgwardt, K.M., Rasch, Gretton MJA.., Kriegel, H.P., et al.: Integrating structured biological data by kernel maximum mean discrepancy. Bioinformatics **22**(14), e49–e57 (2006)

Cormen, T.H., Leiserson, C.E., Rivest, R.L., et al.: Introduction to Algorithms, 3rd edn. MIT Press (2009)

Cramér, H.: On the composition of elementary errors: first paper: mathematical deductions. Scand. Actuar. J. **1928**(1), 13–74 (1928)

Croux, C., Rousseeuw, P.J.: Time-efficient algorithms for two highly robust estimators of scale. In: Computational Statistics. Springer, pp 411–428, (1992)

Curran, J., Hersh, T.: Hotelling. https://CRAN.R-project.org/package=Hotelling, version 1.0-8 (2021)

Cuesta-Albertos, J.A., Fraiman, R., Ransford, T.: Random projections and goodness-of-fit tests in infinite-dimensional spaces. Bull. Braz. Math. Soc. **37**(4), 477–501 (2006)

Friedman, J.H., Rafsky, L.C.: Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests. Ann. Stat. **7**(4), 697–717 (1979)

Fukumizu, K., Bach, F.R., Jordan, M.I.: Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. J. Mach. Learn. Res. **5**, 73–99 (2004)

Fukumizu, K., Gretton, A., Sun, X., et al.: Kernel measures of conditional dependence. In: Advances in Neural Information Processing Systems, pp 489–496, (2008)

Fukumizu, K., Gretton, BA Schölkopf., Sriperumbudur, B.K.: Characteristic kernels on groups and semigroups. In: Advances in Neural Information Processing Systems, pp 473–480, (2009)

Gretton, A., Fukumizu, K., Harchaoui, Z., et al.: A fast, consistent kernel two-sample test. In: Advances in Neural Information Processing Systems 22, (2009)

Gretton, A., Borgwardt, K.M., Rasch, M.J., et al.: A kernel two-sample test. J. Mach. Learn. Res **13**, 723–773 (2012)

Gretton, A., Sejdinovic, D., Strathmann, H., et al.: Optimal kernel choice for large-scale two-sample tests. In: Advances in Neural Information Processing Systems, pp 1205–1213, (2012b)

Gautier L.: rpy2. https://rpy2.github.io/, version 3.5.10 (2010)

Heller, R., Heller, Y.: Multivariate tests of association based on univariate tests. Adv. Neural. Inf. Process. Syst. **29**, 208–216 (2016)

Heller, R., Small, D., Rosenbaum, P.R.: Crossmatch. https://cran.r-project.org/src/contrib/Archive/crossmatch/, version 1.3-1 (2012)

Henze, N., Penrose, M.D.: On the multivariate runs test. Ann. Stat. **1**(27), 290–298 (1999)

Hommel, G.: Tests of the overall hypothesis for arbitrary dependence structures. Biom. J. **25**(5), 423–430 (1983)

Hotelling, H.: The generalization of student's ratio. Ann. Math. Stat. **2**(3), 360–378 (1931)

Huang, C., Huo, X.: An efficient and distribution-free two-sample test based on energy statistics and random projections. arXiv preprint arXiv:1707.04602 (2017)

Hunter, J.D.: Matplotlib: A 2D graphics environment. Comput. Sci. Eng. **9**(3), 90–95 (2007). https://doi.org/10.1109/MCSE.2007.55

Huo, X., Székely, G.J.: Fast computing for distance covariance. Technometrics **58**(4), 435–447 (2016)

Johnson, D.B., Mizoguchi, T.: Selecting the Kth element in $X + Y$ and $X_1 + X_2 + ... + X_m$. SIAM J. Comput. **7**(2), 147–153 (1978)

Knight, W.R.: A computer method for calculating kendall's tau with ungrouped data. J. Am. Stat. Assoc. **61**(314), 436–439 (1966)

Knuth, D.E.: The art of computer programming: fundamental algorithms, 3rd edn. Addison-Wesley, Boston (1997)

Knuth, D.E.: The art of computer programming: sorting and searching, 3rd edn. Addison-Wesley, Boston (1997)

Kolmogorov, A.N.: Sulla determinazione empirica di una legge didistribuzione. Giornale dell'Istituto Italiano degli Attuari **4**, 89–91 (1933)

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

Landau, S., Stahl, D.: Sample size and power calculations for medical studies by simulation when closed form expressions are not available. Stat. Methods Med. Res. **22**(3), 324–345 (2013)

Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**, 50–60 (1947)

Neil, J., Hash, C., Brugh, A., et al.: Scan statistics for the online detection of locally anomalous subgraphs. Technometrics **55**(4), 403–414 (2013)

Panny, W., Prodinger, H.: Bottom-up mergesort—a detailed analysis. Algorithmica **14**(4), 340–354 (1995)

Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: Advances in Neural Information Processing Systems, pp 1177–1184, (2007)

Ramdas, A., Reddi, S.J., Póczos, B., et al.: On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. In: Proceedings of the AAAI Conference on Artificial Intelligence, (2015)

Rosenbaum, P.R.: An exact distribution-free test comparing two multivariate distributions based on adjacency. J. R. Stat. Soc.: Series B (Stat Methodol) **67**(4), 515–530 (2005)

Rousseeuw, P.J., Croux, C.: Alternatives to the median absolute deviation. J. Am. Stat. Assoc. **88**(424), 1273–1283 (1993)

Sejdinovic, D., Sriperumbudur, B., Gretton, A., et al.: Equivalence of distance-based and RKHS-based statistics in hypothesis testing. Ann. Stat. **41**, 2263–2291 (2013)

Shamos, M.I.: Geometry and statistics: problems at the interface. In: In Algorithms and Complexity, Citeseer. Academic Press, Inc , (1976)

Shawe-Taylor, J., Cristianini, N.: Kernel methods for pattern analysis. Cambridge University Press, UK (2004)

Shen, C., Vogelstein, J.T.:The exact equivalence of distance and kernel methods for hypothesis testing. arXiv preprint arXiv:1806.05514 (2018)

Smirnov, N.: Table for estimating the goodness of fit of empirical distributions. Ann. Math. Stat. **19**(2), 279–281 (1948)

Sriperumbudur, B.K., Fukumizu, K., Gretton, A., et al.: Kernel choice and classifiability for RKHS embeddings of probability distributions. In: NIPS, pp 1750–1758, (2009)

Sriperumbudur, B.K., Gretton, A., Fukumizu, K., et al.: Hilbert space embeddings and metrics on probability measures. J. Mach. Learn. Res. **11**, 1517–1561 (2010)

Student,: The probable error of a mean. Biometrika **6**(1): 1–25, (1908)

Székely, G.J., Rizzo, M.L.: Testing for equal distributions in high dimension. InterStat **5**(16.10), 1249–1272 (2004)

Székely, G.J., Rizzo, M.L.: Energy statistics: a class of statistics based on distances. J. Stat. Plann. Infer. **143**(8), 1249–1272 (2013)

Székely, G.J., Rizzo, M.L., Bakirov, N.K.: Measuring and testing dependence by correlation of distances. Ann. Stat. **35**(6), 2769–2794 (2007)

Wei, S., Lee, C., Wichers, L., et al.: Direction-projection-permutation for high-dimensional hypothesis tests. J. Comput. Graph. Stat. **25**(2), 549–569 (2016)

Wilcoxon, F.: Individual comparisons by ranking methods. Biomet. Bull. **1**(6), 80–83 (1945)

Zaremba, W., Gretton, A., Blaschko, M.: B-test: A non-parametric, low variance kernel two-sample test. In: Advances in Neural Information Processing Systems, pp 755–763, (2013)

Zhang, Y., Scheuermann, R.H.: (2020) FR-Match. https://github.com/JCVenterInstitute/FRmatch/

Zhang, Y., Aevermann, B.D., Bakken, T.E., et al.: Fr-match: robust matching of cell type clusters from single cell RNA sequencing data using the friedman-rafsky non-parametric test. Briefings Bioinf. **22**(4), bbaa339 (2021)

Zhao, J., Meng, D.: FastMMD: Ensemble of circular discrepancy for efficient two-sample test. Neural Comput. **27**(6), 1345–1372 (2015)