Precise Change Point Detection using Spectral Drift Detection*

Fabian Hinder, André Artelt†, Valerie Vaquet and Barbara Hammer

Bielefeld University - Cognitive Interaction Technology (CITEC)
Inspiration 1, 33619 Bielefeld - Germany

May 16, 2022

Abstract

The notion of concept drift refers to the phenomenon that the data generating distribution changes over time; as a consequence machine learning models may become inaccurate and need adjustment. In this paper we consider the problem of detecting those change points in unsupervised learning. Many unsupervised approaches rely on the discrepancy between the sample distributions of two time windows. This procedure is noisy for small windows, hence prone to induce false positives and not able to deal with more than one drift event in a window. In this paper we rely on structural properties of drift induced signals, which use spectral properties of kernel embedding of distributions. Based thereon we derive a new unsupervised drift detection algorithm, investigate its mathematical properties, and demonstrate its usefulness in several experiments.

Keywords: Concept Drift · Drift Detection · Data Streams · Non-Parametric Methods · Kernel Methods · Spectral Clustering.

1 Introduction

One fundamental assumption in classical machine learning is the fact that observed data are i.i.d. according to some unknown probability $P_X$, i.e. the data generating process is stationary. Yet, this assumption is often violated as soon as machine learning faces data from the real world such as social media entries or measurements of IoT devices, which are subject to continuous change [2, 26]. Here, concept drift, i.e. changes of the underlying distribution $P_X$, can be caused by seasonal changes, changed demands, ageing of sensors, etc.

The presence of drift can be modeled in different ways. Covariate shift refers to the situation of training and test set having different marginal distributions.

---

*Funding in the frame of the BMBF project TiM, 05M20PBA, and from the VW-Foundation for the project IMPACT is gratefully acknowledged.

†Affiliation with the University of Cyprus
Learning from data streams extends this setting to an unlimited but usually countable stream of observed data [10]. Here, one distinguishes between virtual and real drift, i.e., non-stationarity of the marginal distribution only or also the posterior. Learning technologies for such situations often rely on windowing techniques, and adapt the model based on the characteristics of the data in an observed time window. Active methods explicitly detect drift, usually referring to drift of the classification error, and trigger model adaptation this way, while passive methods continuously adjust the model [7].

In recent years, quite a few approaches on how to deal with concept drift were proposed [7, 21]. They range from non-parametric methods over gradient techniques to ensemble technologies for dealing with streaming data [20]. The precise pinpointing of the time point of a drift event, commonly referred to as drift detection, constitutes an important sub-problem which is used in active learning methods as well as approaches to analyse and understand potentially drifting data streams. A large number of drift detection methods exist, which partially also characterize the overarching type of drift [1, 11]. Many approaches deal with supervised scenarios, aiming for a small interleaved train-test error, and first approaches for their explanation [27]. As drift can induce problems in all kinds of information analysis and processing tasks where label information is not available [12], we focus on the unsupervised case for drift detection. Many unsupervised drift detection schemes rely on a comparison of distributions taken from two consecutive time windows [11]. These techniques can easily lead to false positives; moreover, they are not capable of dealing with consecutive drift events in close proximity [16].

In this contribution, we aim for a different unsupervised drift detection technology which relies on spectral properties of diverging probability distributions rather than their direct comparison based on a distance measure. In targeting this aim, our contribution is twofold: (1) We provide mathematical insights which enable us to link the presence of drift in a data stream to the spectral structure of the induced kernel matrix. (2) Based on this insight, we propose a new robust drift detection algorithm which is capable of reliably detecting drift events in a given data stream. The approach is based on a spectral analysis of kernel matrices and links the problem of drift detection to spectral clustering. As a result we obtain a very precise estimation of the time point of multiple drift events as we will demonstrate in a number of experiments.

The remainder of this work is structured as follows: first (Section 2), we recall the relevant work from literature and define the problem setup. In Section 3 we derive the underlying theory that establishes the regularity properties of the drift induced signals and show its connection to spectral theory (Theorem 1). Thereon we derive an efficient drift detection algorithm (Section 4). In the last section (Section 5) we empirically evaluate the usefulness of our approach for the problem of drift detection.
2 Problem Setup and Related Work

In the usual, time invariant setup of machine learning, one considers a generative process $P_X$, i.e. a probability measure, on $\mathbb{R}^d$. In this context, one views the realizations of $P_X$-distributed random variables $X_1, \ldots, X_n$ as samples. Depending on the objective, learning algorithms try to infer the data distribution based on these samples, or, in the supervised setting, the posterior distribution. We will not distinguish between these settings and only consider distributions in general, subsuming supervised and unsupervised modeling [27].

Many processes in real-world applications are time dependent, so it is reasonable to incorporate time into our considerations. One prominent way to do so, is to consider an index set $T$, representing time, and a collection of probability measures $p_t$ on $\mathbb{R}^d$, indexed over $T$, which may change over time [10]. We will usually assume $T \subset \mathbb{R}$. Drift refers to the fact that $p_t$ varies for different time points, i.e.

$$\exists t_0, t_1 \in T : p_{t_0} \neq p_{t_1}.$$ 

In this context, we consider a sequence of samples $(X_1, T_1), (X_2, T_2), \ldots$, with $X_i \sim p_{T_i}$ and $T_i \leq T_{i+1}$, as a stream. Notice, that we will usually use the shorthand notion drift instead of concept drift. More formally we define:

**Definition 1.** Let $T \subset \mathbb{R}$ be an open (time) interval, and let $\mathcal{X}$ be the (measurable) data space. A drift process on $\mathcal{X}$ over $T$ is a map that assigns a distribution $p_t$ on the data space for every time point $t \in T$ such that $t \mapsto p_t(A)$ is a measurable map for all measurable $A \subset \mathcal{X}$, i.e. a Markov kernel from $T$ to $\mathcal{X}$. We say that $p_t$ has drift if \[\{(s, t) \in T^2 \mid p_s \neq p_t\}\] is not a Lebesgue null-set.

We say that $p_t$ has abrupt drift only if there exist time points $t_1 < \cdots < t_n$, $t_i \in T$, called change points, and probability measures $P_0, P_1, \ldots, P_n$ on $\mathcal{X}$, called concepts, such that $P_i \neq P_{i+1}$ and

$$p_t = \begin{cases} 
P_0 & t < t_1 \\
P_1 & t \in [t_1, t_2) \\
\vdots & \\
P_n & t \geq t_n
\end{cases},$$

up to a Lebesgue null-set.

A relevant problem is to detect drift, i.e. identify all time points $t$ such that $p_t$ and $p_{t+\Delta}$ differ, for some small time interval $\Delta > 0$. If the underlying distributions are known, this can be done using a metric, i.e.

$$t \mapsto d(p_t, p_{t+\Delta}).$$

We are interested in characterizing the shape of this function in empirical observations. As it will turn out, this will provide important information regarding the drift of the underlying system, which enables a robust localization of the
change point in time from observed data. Furthermore, if we take the derivative at \( \Delta = 0 \) this is closely related to the rate of change or drift speed of the stream. As we will show, this is one of the most relevant quantities in drift detection, as many drift detection schemes rely on it in one way or another.

As \( p_t \) are usually not known, most drift detection algorithms which have been proposed in the literature are based on the comparison of sample characteristics of two time windows [11]. The samples within those time windows are then used to estimate the momentary discrepancies between the current underlying distribution of the stream and a reference time frame. Popular drift detection methods differ in the choice of distance measures, their transfer into a statistics, and the allocation of samples to the windows [21]. Here, we will mainly consider the case of two consecutive windows, that are both sliding along the stream. In the literature, a common alternative is to use only one sliding window and compare this to a reference sample; this setup enables the detection of slow drift as well, but it requires a strategy how to chose the reference window: Depending on the specific implementation, this may lead to a refractory state after having detected a drift, where reference samples need to be collected before being capable of detecting the next drift. Notice that the subsequent statements also take more general windowing techniques into account. However, fixed and sliding windows behave in a similar way for the case of abrupt drift, in which we are interested.

The authors of [27] provided a quantification scheme for drift – drift magnitude – defined as

\[
\sigma_{d,t,p}(t) = d(p_{[t-2l,t+l]}, p_{[t-l,t]})
\]

where \( p_{[a,b]} \) denotes the mean distribution in the time interval \([a,b]\) and \( d \) is a distance measure. Notice, that this is the specific instantiation of the original definition by [27], which focuses on consecutive time windows. Assuming the samples arrive with equidistant delay, we may estimate \( \sigma \) based on the last \( 2n_l \) samples, where \( n_l \) is the number of samples in a time period of length \( l \). In [27] \( d \) is chosen as the total variation norm, in [6] Hellinger distance is used, [25] and [16] use the Maximum Mean Discrepancy (MMD) [13], [5] use Kullback-Leibler divergence, [24] use the (feature wise) maximum-norm of the cdfs, and [12] use classification accuracy of a linear model, which is thus related to the total variance and MMD (with linear kernel).

In [16] the theoretical properties of \( \sigma \) are analyzed. The main results are that \( \sigma \) is indeed able to identify drift for a valid distance function by taking on values larger than zero, which also points out the point in time. However, those effects are usually spread out in time. Although the work [16] identifies a characteristic shape of this function in case of isolated, abrupt drift events, a theoretical description of the shape in general is still an open question.

In the next section we investigate this problem for the special case of kernel induced metric MMD. We derive a formula to describe the shape of \( \sigma \) for general drift processes and show its connection to the problem of measuring the rate of change and, in the case of abrupt drift events, spectral clustering.
3 The Shape of the Drift Magnitude

The drift magnitude has been introduced to examine the rate of change of a single or multiple features. We will focus on the case where all features are considered at once, resulting in a single time series per data stream. We will focus on Maximum Mean Discrepancy \[13\] as distance measure, which is defined as

\[
\text{MMD}(P,Q) = \|\mu_P - \mu_Q\|_\mathcal{H},
\]

where \(\mu_P\) and \(\mu_Q\) are kernel embeddings of \(P\) and \(Q\), respectively, and \(\|\cdot\|_\mathcal{H}\) is the norm in the associated Hilbert-space.

Recall that a kernel \(k\) over a space \(X\) is a map \(k : X \times X \rightarrow \mathbb{R}\) that factors through the inner product of a Hilbert \(\mathcal{H}\) of functions from \(X\) to \(\mathbb{R}\), such that \(k(x,\cdot) \in \mathcal{H}\) and \(\langle k(x,\cdot), f \rangle_{\mathcal{H}} = f(x)\) for all \(f \in \mathcal{H}\). In particular, \(k(x,y) = \langle k(x,\cdot), k(y,\cdot) \rangle_{\mathcal{H}}\); here \(k(x,\cdot)\) is usually denoted by \(\varphi(x)\) and called the feature map. Note, that \(\mathcal{H}\) is uniquely determined by this property. Furthermore, given a measure \(P\) on \(X\) we can embed it into \(\mathcal{H}\) by integrating \(\varphi\) with respect to \(P\), i.e.

\[
\mu_P := \varphi(P) := \int \varphi(x) dP(x).
\]

If \(k\) is universal, i.e. every continuous function can be approximated by functions in the subspace of \(\mathcal{H}\) spanned by \(\varphi\), then \(\|P - Q\|_k = \|\mu_P - \mu_Q\|_\mathcal{H}\) defines a norm on the set of all finite measures, which is known as Maximum Mean Discrepancy (MMD).

Since we are mainly interested in the embedding of \(p_t\) for different values of \(t\), we define a map \(K : T \times T \rightarrow \mathbb{R}\) that resembles a kernelized version of the auto-correlation of \(p_t\):

\[
K_{t,s} := \langle \mu_{p_t}, \mu_{p_s} \rangle.
\]

Notice, that integrals commute with the inner product so that it makes sense to consider the mean kernel auto-correlation \(K_{w,w'}\) over time-windows \(w\) and \(w'\). Furthermore, as MMD is the norm of a Hilbert space, we can express it for different time points using the kernel auto-correlation:

\[
\|p_t - p_s\|_k = K_{t,t} - 2K_{t,s} + K_{s,s},
\]

which again holds for time windows, too. There is a close connection between the auto-correlation and the kernel matrix:

**Corollary 1.** Let \(X_1, \ldots, X_n \sim p_w\) i.i.d. and \(Y_1, \ldots, Y_m \sim p_{w'}\) i.i.d. be samples drawn from the stream during the time windows \(w\) and \(w'\), with \(X_i\) and \(Y_j\) independent unless \(X_i = Y_j\) then it holds

\[
\frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} k(X_i, Y_j) \xrightarrow{n,m \to \infty} K_{w,w'},
\]

in probability with rate \(O(\min\{m,n\}^{-1/2})\), assuming \(k\) is bounded.

The notion of the kernel auto-correlation will allow us to derive a general formula for the shape of the drift magnitude \(\sigma\). However, before we do so, notice that instead of using two sliding windows we can actually use any arbitrary weighting scheme:
**Definition 2.** Let \( T' \subset T \) be a measurable set. A *weighting scheme* is a finite, signed stochastic kernel \( w \) from \( T' \) to \( T \), i.e. \( w_t \) is a finite signed measure on \( T \) for all \( t \in T' \), and the map \( t \mapsto w_t(A) \) is measurable for all measurable sets \( A \subset T \).

Notice, that general weighting schemes subsume combinations of fixed, sliding, and growing reference windows with sliding window. The special case of two sliding windows corresponds to the weighting scheme induced by the convolution with some function \( w_0 \), i.e. \( w_t(A) = \int_A w_0(s-t)ds \). By setting \( w_0 = \mathbb{I}_{[-l,0]} - \mathbb{I}_{[-2l,-l]} \) we obtain the sliding window of length \( l \), i.e.

\[
\|w_0 * p_t\|_k^2 := \left\| \int p_s w_0(s-t)ds \right\|_k^2 = \sigma_{\| \cdot \|_k, f, l} (t).
\]

As it turns out a convolution weighting scheme can be particularly well analyzed as we will see later on in Theorem 1. Before we turn to this special case we consider the general notion of a weighting scheme, as it allows us to derive the following lemma which is applicable to a large variety of drift detection schemes: (Due to space restrictions all proofs in this article are contained in the supplemental material.)

**Lemma 1.** Let \( p_t \) be a drift process, \( w \) be a weighing scheme, and \( k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \) be a bounded kernel. Assume \( \mathcal{X} \subset \mathbb{R}^d \) is compact. Then the eigenfuctions \( c_i : \mathcal{X} \to \mathbb{R} \) and eigenvalues \( \lambda_i \in \mathbb{R} \geq 0 \), \( i = 1, \cdots \), of \( K_{t,s} \) always exist. Denote by \( f_i(t) = \int c_i(x)dp_t(x) \). Then it holds

\[
\sum_{i=1}^{\infty} \lambda_i \left( \int f_i(s)dw_t(s) \right)^2 = \left\| \int p_s dw_t(s) \right\|_k^2 \quad \text{in } L^2.
\]

Thus, many algorithms consider squared, overlaying, weighted functions that are derived from the drift process. Therefore, it is reasonable to assume that intricate patterns that occur in drift induced signals are a result of the used weighting; indeed, in [16] it is shown that abrupt drift events are distributed in time by the drift magnitude. In the following we will make this more explicit by restricting ourselves to the case of a convectional weighting scheme and consider drift in the formal framework of (generalized) derivatives.

### 3.1 The Shape of the Drift Magnitude for Sliding Windows

Notice, that Lemma 1 shows a tight connection between general drift and drift that is induced by mixing components, which refers to the fact that there exist probability measures \( P_1, \ldots, P_n \) and functions \( f_1, \ldots, f_n : T \to \mathbb{R}_{\geq 0} \) with \( \sum_i f_i(t) = 1 \) such that

\[
p_t = \sum_{i=1}^{n} f_i(t)P_i.
\]
In the kernel space, the “components” $P_i$ are provided by the eigenfunctions $c_i$ and, since eigenfunctions are orthonormal, the roles of the $f_i$ coincide. Mixing components are interesting as they are easy to analyze: If we denote by $w_l$ and $h_l$ the windowing and shape function [16, Theorem 1], respectively, then it holds

$$\sigma_{||.||_{k,l,p}}(t) = \left|\left| \sum_{i=1}^{n} w_l * f_i(t) P_i \right|\right|_{k}^2 = \sum_{i,j} \langle h_l * f'_i(t) \cdot (h_l * f'_j(t)) \langle \mu_{P_i}, \mu_{P_j} \rangle_H \rangle.$$

As we will see, this idea carries over to general drift processes, assuming we make use of MMD as metric. Indeed, it is the most fundamental quantity in the context of drift detection using sliding windows. To make this precise we need to generalize the notion of differentiability to drift processes. Intuitively, a drift process is differentiable if any derived quantity depends differentiable on time. Formally we define:

**Definition 3.** Let $p_t$ be a drift process. We say that $p_t$ is (distributional/weak) differentiable if $t \mapsto p_t(A)$ is (distributional/weak) differentiable for all measurable sets $A \subset \mathcal{X}$.

**Remark 1.** Notice that

1. for $\mathcal{X} = \mathbb{R}^d$ it holds that $p_t$ is (distributional/weak) differentiable if and only if $t \mapsto \int f(x) dp_t(x)$ is (distributional/weak) differentiable for all measurable, bounded continuous, or bounded smooth functions $f: \mathcal{X} \to \mathbb{R}$. In particular, if $p_t$ is (distributional/weak) differentiable then the functions $f_i$ from Lemma 1 are (distributional/weak) differentiable.

2. mixing processes are (distributional/weak) differentiable if all $f_i$ are (distributional/weak) differentiable.

3. drift processes with abrupt drift only are distributional differentiable.

Using this definition, we can now relate the drift magnitude to the notion of derivatives. As it turns out, the signals obtained by the drift magnitude are essentially contorted derivatives:

**Theorem 1.** Let $p_t$ be a (distributional/weak) differentiable drift process, $w: \mathbb{R} \to \mathbb{R}$ be a (weighting) function, and $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a bounded kernel, i.e. $k(x, x) < C$ for all $x \in \mathcal{X}$. Denote by $W$ a antiderivative (primitive integral) of $w$. Assume $\mathcal{X}$ is compact, $\mathcal{T} \subset \mathbb{R}$ is a bounded domain, and $w$ and $W$ have bounded support. Then the eigenfunctions of $K_{w,t}$ (considered as an integral operator) have (distributional/weak) derivatives $f'_i$ (with eigenvalues $\lambda_i$) and it holds

$$\sum_{i=1}^{\infty} \lambda_i (W * f'_i)^2(t) = ||w * p_t||_K^2$$

$^{1}$We recall the relevant definitions in the supplement.
Figure 1: Four stages of the Spectral Drift Detection Method. Red lines (a,b,c) mark drift events. (d) shows R2-score for Decision Trees with varying number of leaves (1-5); as can be seen optimal score is obtained at 3 leaves/drifts.

in $L^2$, where $\cdot * \cdot$ denotes the convolution operator.

In particular, if $p_t$ has abrupt drift only, with change points $t_1, \ldots, t_n$ and concepts $P_0, P_1, \ldots, P_n$ then the expression simplifies to

$$\sum_{i=1}^{\infty} \left( \sum_{j=1}^{n} s_{i,j} W(t - t_j) \right)^2 = \| w * p_t \|_k^2,$$

with $\sum_{i=1}^{\infty} s_{i,j}^2 = \| P_j - P_{j-1} \|_k^2$. Furthermore, if the length of the support of $W$ is smaller than the minimal distances between the $t_i$, i.e. if supp $W \subset [a, a+l]$ and $l < \min \{t_{i+1} - t_i \mid i = 1, \ldots, n-1\}$, then only one summand is active at once and the sum commutes with square roots:

$$\sum_{j=1}^{n} \| P_j - P_{j-1} \|_k W(t - t_j) = \| w * p_t \|_k.$$

Notice, that the last formulation is exactly the statement of [16, Corollary 1], observing that the antiderivative of the windowing function $I_{[-l,0]} - I_{[-2l,-l]}$ is exactly the shape-function $h_l$ from [16, Theorem 1]. Furthermore, Theorem 1 establishes a direct connection between the drift magnitude and the rate of change in terms of the derivatives of the eigenfunctions of the kernel autocorrelation.

Since a large variety of drift detection algorithms can be analyzed using Lemma 1, including D3 [12], KS-Win [24], MMDDDM [25], and ShapeDDM [16], we observe that they all analyze a weighted sum of smoothed out eigenfunctions. The main idea of spectral drift detection is to instead analyze the eigenfunctions directly. In the next section we will derive a new drift detection algorithm by adapting spectral clustering [22] to the task at hand.

4 Spectral Drift Detection Method

The core observation that leads to the Spectral Drift Detection Method (SDDM) is the following: If $p_t$ has abrupt drift only, then a kernel matrix obtained from
samples drawn from the drift process takes on the shape of a block matrix, where the boundaries of the blocks corresponds to the change points and the blocks correspond to samplings from the mean auto-correlation. We have illustrated this in Fig. 1. Formally, we obtain the following statement:

**Theorem 2.** Let \( p_t \) be a drift process with abrupt drift only at \( t_1 < \cdots < t_n \), on \( T = [t_0, t_{n+1}] \), and let \( k : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \) be a kernel. Then there exists a matrix \( \tilde{K} \in \mathbb{R}^{(n+2) \times (n+2)} \) such that

\[
K_{t,s} = \sum_{i,j=0}^{n+1} \mathbb{I}_{[t_i, t_{i+1}]}(t) \tilde{K}_{i,j} \mathbb{I}_{[t_j, t_{j+1}]}(s).
\]

And it holds that the eigenvalues of \( K \) and \( \tilde{K} \) coincide and if \( v_l \) is the \( l \)-th eigenvector of \( \tilde{K} \), then

\[
f_l(t) = \sum_{i=0}^{n+1} \mathbb{I}_{[t_i, t_{i+1}]}(t) \frac{(v_l)_i}{t_{i+1} - t_i}
\]

is the \( l \)-th eigenfunction of \( K \).

As stated above, SDDM directly analyzes the eigenvectors of the kernel matrix. This allows us to connect it to other methods based on MDD, which usually consider \( \hat{\text{MMD}} = w^\top \hat{K} w \), where \( \hat{K} \) is the kernel matrix and \( w = (-1/n_0, \ldots, -1/n_0, 1/n_1, \ldots, 1/n_1) \) with \( n_0 \) and \( n_1 \) the length of the reference and current window, respectively. It is easy to see that sliding along the stream (MMD-DDM, Shape-DDM) or using auto-cuts (comparable to ADWIN) corresponds to an optimization problem over those vectors \( w \), which essentially approximates, and in case of a single drift event is equivalent to, the computation of the larges eigenfunction of \( K_{s,t} \). Furthermore, if we consider \( p_t \) as a mixing problem, as suggested by Lemma 1, then the eigenvectors of the kernel matrix are approximations of the mixing functions \( f_i(t) \).

Due to the connection between the kernel matrix of streaming data and the kernel auto-correlation function, we obtain the drift detection algorithm by applying a modified version of Spectral Clustering [22] to the kernel matrix. Intuitively speaking, we form clusters of those time points with the same distribution. Recall that the spectral clustering algorithm is defined as follows:

1. Compute a similarity matrix \( K \) from data points \( x_1, \ldots, x_n \)
2. Compute the row sum matrix \( D = \text{diag}(K1) \)
3. Compute the (normalized) Laplacian \( L = D - K \) (\( L = I - D^{-\frac{1}{2}}KD^{-\frac{1}{2}} \) or \( L = I - D^{-1}K) \)
4. Compute the first \( k \) eigenvectors \( v_1, \ldots, v_k \) of \( L \) (eigenvectors with smallest eigenvalues)
5. Apply a partition algorithm, e.g. \( k \)-means, to the rows of the eigenvectors, i.e. \( \{(v_{1j}, \cdots, v_{kj})^\top \mid j = 1, \cdots, n\} \)
A common choice for the partition algorithm is $k$-means. However, as we want to find clusters of consecutive time points we need another partitioning method. To do so recall that $k$-means tries to find clusters $C$ such that the inner cluster variance is minimized. In addition, we want that the corresponding time points form time intervals, i.e. we want to find time intervals (correspond to clusters) such that the inner variance of the associated samples is minimized. Notice, that this is exactly the objective of decision tree regression with variance reduction, i.e. we replace Step 5 in the Spectral Clustering algorithm by:

5’. Train a decision tree to predict $t_j \mapsto (v_{1j}, \cdots, v_{kj})$, with $t_j$ the arrival time of $x_j$. The boundary points of the leafs correspond to the drift events.

Notice, that the plain CART algorithm is not suited to find optimal trees, as it produces far too many leaf nodes. To solve this issue we suggest to limit the number of leaf nodes to the number of expected drift events. To obtain the number of drift events, observe that the mean of $v_{ij}$ is the same for all $t_j$ between two drift events and different otherwise. Thus, the number of change points is equivalent to the optimal number of leafs, due to the bias-variance tradeoff. Therefore, we can obtain the number of drift events via cross-validation. We illustrated an example of the score for different numbers of leafs in Fig. 1d. This completes the derivation of the Spectral Drift Detection Method, which is summarized in Algorithm 1.

4.1 Implementation Details and Algorithmic Properties

We will now discuss some of the aspects of SDDM in more detail.

**MomentTree-Kernel** Although any kernel can be used for SDDM, we suggest to make use of a kernel based on MomentTrees [18] which seems to provide a better distance measure than the commonly used Gauss-kernel [19].

MomentTrees is a method for decision tree based conditional density estimation; in the context of drift they are trained to predict the arrival time $t$ based on the sample $x$ [19], i.e. they learn $x \mapsto t$. The idea is derived from the observation that, by using conditional density estimation, the data space can

![Figure 2: Comparison of MomentTree (MT) and Gauss (RBF) kernel matrix on ForestCovertype dataset (ABA, $|B| = 100$; see Section 5) and first eigenvector of both kernels matrices. MT shows significant difference between drifts.](image)
Algorithm 1 Spectral Drift Detection Method

1: procedure SDDM: Spectral Drift Detection Method((x_i) data stream, n_{eigen} number of eigenvectors, n_{itr} number of iterations in cross-validation, k_{max} maximal number of drifts)
2: Initialize Window W ← []
3: while Not at end of stream (x_i) do
4:     W ← W + [x_i] \quad \triangleright Add new sample
5:     if |W| > n_{max} then
6:         \text{pop}(W) \quad \triangleright Drop oldest sample
7:     end if
8:     if |W| > n_{min} then
9:         K ← \text{ComputeKernel}(W) \quad \triangleright Compute similarity matrix
10:        D'_{ii} ← \left( \sum_{j=1}^{W} K_{ij} \right)^{-\frac{1}{2}}
11:        L = I - D'KD'
12:        V ← \text{eigen}(L, n_{eigen}) \quad \triangleright Compute n_{eigen} first eigenvectors
13:        k ← \text{FindNumberOfLeafsByCrossValidation}(n_{itr}, k_{max}, V)
14:        if k > 1 then
15:            t ← \text{TrainTreeWithLeafs}(V, k)
16:            d_1, \ldots, d_k ← \text{ExtractDecisionThresholds}(t)
17:            Alert drift at d_1, \ldots, d_k
18:        end if
19:     end if
20: end while
21: end procedure

be segmented into regions with the same drift behavior [17]. Due to this fact, the trees will find a segmentation of the data space that provides a particularly good binning to measure the distance between p^t and p^s [19].

To obtain a kernel we use tree similarities [4], i.e. we train RandomForest of MomentTrees on the current window, the similarity of two points is given by the ratio of trees for which both point belong to the same leaf region. We illustrated the benefit of using MomentTree-kernels over Gauss-kernels in Fig. 2.

Decision Threshold / p-Value In contrast to many other drift detection methods, SDDM is not based on a statistical test to determine whether or not a drift occurred at certain time. Therefore, we do not need a threshold p- or \alpha-value which is usually hard to estimate. Instead, SDDM uses a decision process, which is based on cross-validation, to determine the number of drift points. Although the cross-validation needs some parameters (number of iterations, train-test-split ratio), the method turns out to be robust in a large spectrum of choices.
Batch and Offline-Mode. Notice, that SDDM detects all drift events within a given window. Therefore, it can be applied in offline mode, i.e. we only run lines 9 to 18 in Algorithm 1. This is also beneficial for applying SDDM in an online setup as it involves several, computational expensive tasks as the training of the MomentTrees, the computation of the eigenvectors, and performing the cross-validation. Those issues can be solved by running lines 9 to 18 in a batch mode, i.e. instead of running them every time a new sample is received, we only execute them after a certain number of new samples are received.

Postprocessing. As SDDM will alert the same drift events as long as they are present in the current window, it is required to filter out events that are found multiple times. Though, the selection usually pinpoints the exact same time-stamp during several runs, we postprocess the selection using Ward clustering. Furthermore, as false alarms usually only occur in a single batch we can reduce the number of false alarms by considering the drift events that are found in most batches (that can potentially contain them) only.

5 Experiments

We performed empirical experiments to evaluate capability of our method to detect abrupt drift and determine the time point of the drift.

We used standard drift benchmarks: “Rotating Hyperplane” [23], “STAGGER” [9] and “RandomRBF” [23]. And the following real world datasets: “Electricity Market Prices” [15], “Forest Covertype” [3], and “Nebraska Weather” [8]. If a dataset provides a label, it is considered as an additional feature, thereby real drift is turned into distributional drift. To control the concepts obtained from the real world datasets, we split the datasets into time windows – which then correspond to the concepts – and then selecting samples randomly from those, thereby removing any unwanted drift during one concept. Using these concepts (either from the artificial or the real world dataset) we create data streams with a length of 1,600 samples each, which consist of three segments (Warmup, Drift, and Cool down); the first (W), and last segment (O) contain

Figure 3: Results on Nebraska Weather dataset (ABC, $|B| = 100$) over 150 runs. x-axis represents time, y-axis and color represent method (and run), crosses mark found drift events, red lines mark true drift events.
Table 1: Results over 150 runs. Table shows mean $\beta$-score ($\beta = 0.5$) and standard deviation. Numbers below stream types are B-length and maximal delay.

| Dataset          | Method     | $\beta$-score | 25/12 | 50/25 | 25/12 | 50/25 |
|------------------|------------|---------------|-------|-------|-------|-------|
| Electricity      | SDDM (MT)  | 0.87±0.27     | 0.20±0.36 | 0.65±0.40 | 0.86±0.22 | 0.95±0.10 |
| Market Prices    | SDDM (RBF)| 0.50±0.46     | 0.00±0.00 | 0.01±0.07 | 0.64±0.36 | 0.67±0.34 |
|                  | D3         | 0.17±0.28     | 0.09±0.16 | 0.13±0.17 | 0.29±0.15 | 0.28±0.16 |
|                  | HDDDM      | 0.39±0.47     | 0.00±0.00 | 0.01±0.07 | 0.18±0.23 | 0.25±0.24 |
|                  | KSDD       | 0.09±0.09     | 0.07±0.08 | 0.14±0.10 | 0.18±0.09 | 0.21±0.10 |
|                  | MMDDDM     | 0.14±0.10     | 0.02±0.06 | 0.14±0.16 | 0.17±0.08 | 0.24±0.09 |
|                  | ShapeDDM   | 0.88±0.27     | 0.00±0.04 | 0.03±0.11 | 0.37±0.21 | 0.47±0.09 |
| Forest           | SDDM (MT)  | 0.92±0.15     | 0.69±0.32 | 0.89±0.13 | 0.66±0.24 | 0.87±0.19 |
| Covertype        | SDDM (RBF)| 0.03±0.16     | 0.00±0.00 | 0.00±0.00 | 0.00±0.00 | 0.01±0.06 |
|                  | D3         | 0.44±0.27     | 0.21±0.18 | 0.26±0.15 | 0.29±0.15 | 0.31±0.12 |
|                  | HDDDM      | 0.28±0.38     | 0.03±0.11 | 0.12±0.21 | 0.17±0.21 | 0.33±0.20 |
|                  | KSDD       | 0.06±0.02     | 0.07±0.03 | 0.09±0.03 | 0.09±0.04 | 0.10±0.03 |
|                  | MMDDDM     | 0.17±0.09     | 0.11±0.12 | 0.29±0.15 | 0.19±0.13 | 0.21±0.08 |
|                  | ShapeDDM   | 0.98±0.07     | 0.02±0.10 | 0.22±0.25 | 0.47±0.11 | 0.47±0.11 |
| Random RBF       | SDDM (MT)  | 0.93±0.16     | 0.20±0.35 | 0.78±0.31 | 0.56±0.22 | 0.85±0.23 |
|                  | SDDM (RBF)| 0.06±0.23     | 0.00±0.00 | 0.01±0.08 | 0.01±0.07 | 0.14±0.27 |
|                  | D3         | 0.27±0.30     | 0.12±0.17 | 0.18±0.19 | 0.15±0.16 | 0.28±0.16 |
|                  | HDDDM      | 0.21±0.33     | 0.01±0.08 | 0.03±0.11 | 0.11±0.17 | 0.07±0.17 |
|                  | KSDD       | 0.09±0.06     | 0.08±0.07 | 0.13±0.07 | 0.12±0.08 | 0.16±0.07 |
|                  | MMDDDM     | 0.21±0.10     | 0.10±0.13 | 0.22±0.09 | 0.23±0.12 | 0.21±0.09 |
|                  | ShapeDDM   | 0.97±0.10     | 0.01±0.08 | 0.14±0.22 | 0.48±0.09 | 0.48±0.05 |
| Rotating Hyperplane | SDDM (MT)  | 0.24±0.42     | 0.00±0.04 | 0.00±0.00 | 0.18±0.23 | 0.20±0.25 |
|                  | SDDM (RBF)| 0.77±0.21     | 0.00±0.00 | 0.00±0.00 | 0.70±0.29 | 0.75±0.25 |
|                  | D3         | 0.08±0.23     | 0.03±0.11 | 0.04±0.13 | 0.08±0.16 | 0.11±0.18 |
|                  | HDDDM      | 0.03±0.16     | 0.01±0.07 | 0.02±0.09 | 0.00±0.04 | 0.03±0.11 |
|                  | KSDD       | 0.04±0.10     | 0.06±0.10 | 0.07±0.11 | 0.04±0.08 | 0.07±0.10 |
|                  | MMDDDM     | 0.17±0.09     | 0.03±0.09 | 0.05±0.11 | 0.17±0.08 | 0.21±0.10 |
|                  | ShapeDDM   | 0.91±0.15     | 0.00±0.04 | 0.02±0.09 | 0.43±0.15 | 0.46±0.12 |
| STAGGER          | SDDM (MT)  | 0.88±0.30     | 0.46±0.47 | 0.66±0.43 | 0.68±0.33 | 0.72±0.38 |
|                  | SDDM (RBF)| 0.71±0.33     | 0.11±0.28 | 0.24±0.33 | 0.31±0.33 | 0.42±0.30 |
|                  | D3         | 0.27±0.31     | 0.17±0.18 | 0.25±0.18 | 0.18±0.17 | 0.28±0.18 |
|                  | HDDDM      | 0.24±0.33     | 0.01±0.08 | 0.05±0.13 | 0.06±0.14 | 0.16±0.20 |
|                  | KSDD       | 0.28±0.28     | 0.14±0.17 | 0.17±0.17 | 0.16±0.17 | 0.20±0.20 |
|                  | MMDDDM     | 0.18±0.07     | 0.16±0.15 | 0.24±0.13 | 0.19±0.10 | 0.20±0.09 |
|                  | ShapeDDM   | 0.94±0.12     | 0.06±0.17 | 0.16±0.22 | 0.24±0.25 | 0.28±0.24 |
| Nebraska Weather | SDDM (MT)  | 0.79±0.34     | 0.14±0.30 | 0.49±0.43 | 0.46±0.31 | 0.91±0.13 |
|                  | SDDM (RBF)| 0.07±0.25     | 0.03±0.16 | 0.10±0.26 | 0.49±0.39 | 0.80±0.21 |
|                  | D3         | 0.06±0.18     | 0.12±0.17 | 0.18±0.18 | 0.23±0.17 | 0.24±0.16 |
|                  | HDDDM      | 0.01±0.08     | 0.00±0.04 | 0.02±0.10 | 0.00±0.00 | 0.09±0.19 |
|                  | KSDD       | 0.06±0.05     | 0.07±0.06 | 0.11±0.05 | 0.10±0.04 | 0.12±0.04 |
|                  | MMDDDM     | 0.05±0.10     | 0.06±0.12 | 0.12±0.12 | 0.18±0.09 | 0.24±0.10 |
|                  | ShapeDDM   | 0.55±0.48     | 0.01±0.07 | 0.03±0.12 | 0.17±0.23 | 0.18±0.24 |
500 + \Delta and 600 − \Delta samples, respectively, and continue the neighbouring concept from the middle segment (D), where \Delta is a randomly chosen number in range between 0 and 100. The middle segment (D) is 500 samples long and contains 2 or 3 concepts in the ordering: AB, ABA, or ABC. In case AB, both concepts contain equally many samples. In cases ABA and ABC, the length of concept B is varied (|B| ∈ \{25, 50, 75, 100\}) and the other two concepts are of equal length. Every stream is normalized before running the experiments.

We compare our proposed method spectral drift detection (with MomentTree-kernel) (SDDM (MT)) against the shape based drift detector (ShapeDDM) [16], which matches the drift magnitude against a certain shape, the usual MMD drift detection (MMD-DDM) [13, 25], which applies the MMD two sample test at every time point, the Hellinger distance drift detection method (HDDDM) [6], which computes the Hellinger distance using bins of the marginal distribution, Kolmogorov-Smirnov Windowing (KS-Win) [24], which applies the Kolmogorov-Smirnov test feature wise, and D3 [12], which is one of the state of the art drift detectors for unsupervised drift detection. We also consider SDDM with Gauss-kernel (SDDM (RBF)). As far as given we used default parameters for all methods; if not present or determined, we choose the optimal parameters with respect to 50 independent runs. In particular, for all methods except SDDM we selected a shift of the detected drift points to optimally align them with the true drift events to compensate for potential, inherent delays.

We measure the quality of a method using the \(\beta\)-score [16], which is defined as \(\frac{tp}{p + \beta fp}\) where \(p\) is the number of all drifts, \(tp\) the number of true positives, and \(fp\) the number of false positives. Here, the closest drift alert to the actual drift event is considered as a true positive if it is within range of the maximal delay, otherwise it is considered as a false positive. Every other detection is a false positive. We choose the maximal delay as half the length of concept B (in the cases ABA and ABC) so that is is clear which drift is detected. Part of the results are shown in Fig. 3 and Table 1, significantly best results (\(t\)-test, \(p = 0.001\)) are marked bold face (additional tables and figures are provided in the supplement).

As can be seen, SDDM (MT) outperforms all other methods on nearly all datasets. In particular, in the case of multiple concepts (ABA, ABC) SDDM performs significantly better then any other method. An exception is ShapeDDM which performs quite well in the case AB. Furthermore, SDDM (RBF) also shows suitable performance on some datasets. This implies that the spectral method itself produces decent performance but requires a sufficient kernel to work properly. Furthermore, SDDM (MT and RBF) and ShapeDDM produce relatively few alarms (in median 1 or less false alarms per run), in particular, in cases where they perform not that good. This is in strong contrast to other methods like MMD-DDM or KS-Win which usually produced in median 10-15 false alarms (and up to 30 in median).
6 Conclusion and Future Work

In this contribution we considered the shape and structure of drift induced signals, obtained by MMD. We gave a complete description, in case the signals are induced by sliding windows, by giving a direct connection to the rate of change in terms of derivatives. We also showed that this problem is closely related to the eigenvectors of the kernel matrix. We used this insight to construct a new drift detection method SDDM which is based on a segmentation of the eigenvectors. We compared SDDM to several other unsupervised drift detection methods on several datasets and showed it superiority. We also introduced the concept of the MomentTree-kernel, which showed high potential in context of several dataset.

So far, our method is limited to abrupt drift, we hope to extend it to gradual drift in the future.

References

[1] S. Aminikhanghahi and D. J. Cook. A survey of methods for time series change point detection. *Knowl. Inf. Syst.*, 51(2):339–367, May 2017.

[2] A. Bifet and J. Gama. Iot data stream analytics. *Ann. des Télécomm.*, 75(9-10), 2020.

[3] J. A. Blackard, D. J. Dean, and C. W. Anderson. Covertype data set, 1998.

[4] L. Breiman and A. Cutler. Random forests manual v4. In *Technical report*. UC Berkel, 2003.

[5] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi. An information-theoretic approach to detecting changes in multidimensional data streams. *Interfaces*, 01 2006.

[6] G. Ditzler and R. Polikar. Hellinger distance based drift detection for nonstationary environments. In *2011 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments, CIDUE 2011, Paris, France, April 13, 2011*, pages 41–48, 2011.

[7] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar. Learning in nonstationary environments: A survey. *IEEE Comp. Int. Mag.*, 10(4):12–25, 2015.

[8] R. Elwell and R. Polikar. Incremental learning of concept drift in nonstationary environments. *IEEE Transactions on Neural Networks*, 22(10):1517–1531, Oct 2011.

[9] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. volume 8, pages 286–295, 09 2004.
[10] J. a. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37, Mar. 2014.

[11] I. Goldenberg and G. I. Webb. Survey of distance measures for quantifying concept drift and shift in numeric data. *Knowl. Inf. Syst.*, 60(2):591–615, 2019.

[12] Ö. Gözüaçık, A. Büyükçakır, H. Bonab, and F. Can. Unsupervised concept drift detection with a discriminative classifier. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2365–2368, 2019.

[13] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 513–520, 2006.

[14] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf. Covariate shift and local learning by distribution matching, pages 131–160. MIT Press, Cambridge, MA, USA, 2009.

[15] M. Harries, U. N. cse tr, and N. S. Wales. Splice-2 comparative evaluation: Electricity pricing. Technical report, 1999.

[16] F. Hinder, J. Brinkrolf, V. Vaquet, and B. Hammer. A shape-based method for concept drift detection and signal denoising. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 01–08, 2021.

[17] F. Hinder, B. Hammer, and M. Verleysen. Concept drift segmentation via kolmogorov trees. In *Proceedings of the ESANN, 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2021.

[18] F. Hinder, V. Vaquet, J. Brinkrolf, and B. Hammer. Fast non-parametric conditional density estimation using moment trees. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, 2021.

[19] F. Hinder, V. Vaquet, and B. Hammer. Suitability of different metric choices for concept drift detection. *arXiv preprint arXiv:2202.09486*, 2022.

[20] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Wozniak. Ensemble learning for data stream analysis: A survey. *Inf. Fusion*, 37, 2017.

[21] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2018.
[22] U. V. Luxburg. A tutorial on spectral clustering, 2007.

[23] J. Montiel, J. Read, A. Bifet, and T. Abdessalem. Scikit-multiflow: A multi-output streaming framework. Journal of Machine Learning Research, 19(72):1–5, 2018.

[24] C. Raab, M. Heusinger, and F.-M. Schleif. Reactive soft prototype computing for concept drift streams. Neurocomputing, 416:340–351, 2020.

[25] S. Rabanser, S. Günnemann, and Z. Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019.

[26] S. Tabassum, F. S. F. Pereira, S. Fernandes, and J. Gama. Social network analysis: An overview. Wiley Interdiscip. Rev. Data Min. Knowl. Discov., 8(5), 2018.

[27] G. I. Webb, L. K. Lee, F. Petitjean, and B. Goethals. Understanding concept drift. CoRR, abs/1704.00362, 2017.
A Proofs for Section 3 (The Shape of the Drift Magnitude)

Lemma 1. As $k$ is a kernel on a compact space we may apply Mercers theorem to obtain a representation by means of the eigenfunctions of $k$:

$$k(x, y) = \sum_{i=1}^{\infty} \lambda_i c_i(x)c_i(y),$$

with $\lambda_i$ absolutely summable. The lemma is now a simple computation:

$$\left\lVert \int p_s dw_t(s) \right\rVert_k^2 = \left\langle \int \int \phi(x) dp_s(x) dw_t(s), \int \int \phi(y) dp_{s'}(y) dw_t(s') \right\rangle$$

$$= \int \int \int \int \phi(x,y) dp_s(x) dw_t(s) dp_{s'}(y) dw_t(s')$$

$$= \sum_{i=1}^{\infty} \lambda_i c_i(x)c_i(y) dp_s(x) dw_t(s) dp_{s'}(y) dw_t(s')$$

$$= \sum_{i=1}^{\infty} \lambda_i \left( \int c_i(x) dp_s(x) dw_t(s) \cdot \int c_i(y) dp_{s'}(y) dw_t(s') \right)$$

$$= \sum_{i=1}^{\infty} \lambda_i \left( \int f_i(s) dw_t(s) \right)^2,$$

whereby $!$ holds since $\lambda_i$ is absolute summable and $k$ and thus $c_i$ are bounded. 

A.1 Distributions

Definition 4. Let $\Omega \subset \mathbb{R}^d$ an open, not empty subset. Denote by $D(\Omega)$ the space of all test functions, i.e. smooth functions with compact support, equipped with the topology such that $\phi_n \to \phi$ in $D$ if and only if there exists a compact set $K$ such that $\text{supp} \phi_n \subset K$, $\text{supp} \phi \subset K$ and $\partial^\alpha \phi_n \to \partial^\alpha \phi$ uniformly for all $\alpha$. The space of distributions on $\Omega$ is then given as dual space of $D(\Omega)$; we refer to the elements of this space as distributions.

There is a canonical embedding from the space of locally integrable functions $L^1_{\text{loc}}(\mathbb{R}^d)$ into the space of distributions $f \mapsto T_f$ which is induced by $T_f(\phi) = \int f(x) \phi(x) dx$ where $\phi$ is a test function. We call the distributions that can be represented this way regular distributions.

One important distribution, that is not a regular distribution, is the Dirac-impulse which is induced by $\delta : \phi \mapsto \phi(0)$. 

18
Distributions are relevant for our problem as they admit a notion of derivative that is not available if we consider functions only.

**Definition 5.** Let $\Omega \subset \mathbb{R}^n$ and $\alpha \in \mathbb{N}_0^n$. We say that a distribution $T$ on $\Omega$ is \textit{(distributional) $\alpha$-differentiable} iff there exists a distribution $T'$ such that
\[ T'\phi = (-1)^{|\alpha|} T(D^\alpha \phi) \]
for all test functions $\phi$; here $D^\alpha$ denotes the total derivative with respect to $\alpha$ and $|\alpha|$ the degree of $\alpha$. We then refer to $T'$ as the $\alpha$-th derivative of $T$ and denote it by $D^\alpha T$ (or $T'$ if $n = 1$).

Note, that $T'$ is uniquely determined by this property so that the notation is valid.

**Definition 6.** Let $\Omega \subset \mathbb{R}^n$, $\alpha \in \mathbb{N}_0^n$ and $f \in L^1_{\text{loc}}(\Omega)$ be a locally integrable function. We say that $f$ is \textit{weak $\alpha$-differentiable}, iff there exists a locally integrable function $f' \in L^1_{\text{loc}}(\Omega)$ such that $D^\alpha T_f = T_{f'}$. We refer to $f'$ as the $\alpha$-th weak derivative of $f$ and denote it by $D^\alpha f$ (or $f'$ if $n = 1$).

Notice, that $D^\alpha f$ is uniquely determined by the property and that, if $f$ is $\alpha$-differentiable in the classical sense then the classical and weak derivative coincide.

A very important example of a function that is neither differentiable nor weak differentiable but distributional differentiable is the indicator function $I_{(-\infty,0]}(x)$ which is 1 if $x$ is smaller or equal to 0 and 0 otherwise. Its first derivative is given by the Dirac-impulse, i.e. $I'_{(-\infty,0]} = -\delta$.

This function is of particular interest for our purposes, as we may write a drift process with abrupt drift only, with change points $t_1 < \cdots < t_n$ and concepts $P_0, P_1, \ldots, P_n$ as
\[ p_t = \sum_{i=0}^n I_{(t_i, t_{i+1}]} P_i, \]
where $t_0 = -\infty$ and $t_{n+1} = \infty$. It is now obvious that the (distributional) derivative of $p_t$ is given by
\[ p'_t = \sum_{i=1}^n \delta_{t_i}(P_i - P_{i-1}), \]
where $\delta_t : \phi \mapsto \phi(t)$ denotes the shifted Dirac-impulse.

**Definition 7.** Let $T, T_1, T_2$ be distributions on $\mathbb{R}^d$. The \textit{convolution} of $T$ with a test function $\phi$ is defined as $(T * \phi)(x) = T(\phi(x - \cdot))$. Notice, that this is a test function.

If at least one of $T_1$ or $T_2$ have bounded support, then the \textit{convolution of distributions} is the distribution $T_1 * T_2$ defined by $(T_1 * T_2) * \phi = T_1 * (T_2 * \phi)$ for all test functions $\phi$. 

19
Notice, that the commonly known rule for derivatives of convolutions holds for the convolution of a distribution and a test function, i.e.

\[ D^{\alpha}(T \ast \phi) = (D^{\alpha}T) \ast \phi = T \ast (D^{\alpha} \phi), \]

assuming \( D^{\alpha}T \) exists. This statement can be extended to distributions:

**Lemma 2.** Let \( T_1, T_2 \) be two \( \alpha \)-differentiable distributions with bounded support, then it holds

\[ D^{\alpha}(T_1 \ast T_2) = (D^{\alpha}T_1) \ast T_2 = T_1 \ast (D^{\alpha}T_2). \]

In particular, let \( f, g : \mathbb{R} \to \mathbb{R} \) be functions and let \( f \) be distributional differentiable and \( G \) be an antiderivative of \( g \). Assume that \( g \) and \( G \) have bounded support, then it holds

\[ f \ast g = T_f' \ast G. \]

**Proof.** Let \( \phi \) be any test function. It holds

\[
\begin{align*}
D^{\alpha}((T_1 \ast T_2) \ast \phi) &= D^{\alpha}(T_1 \ast (T_2 \ast \phi)) \\
&= T_1 \ast (D^{\alpha}(T_2 \ast \phi)) \\
&= T_1 \ast ((D^{\alpha}T_2) \ast \phi) \\
&= (T_1 \ast (D^{\alpha}T_2)) \ast \phi.
\end{align*}
\]

For the other equality use that \( T_1 \ast T_2 = T_2 \ast T_1 \). For the second statement use that \( G' = g \) and consider \( T_f \) and \( T_g \).

**Theorem 1.** Using Lemma 1 we can represent

\[ K_{s,t} = \sum_{i=1}^{\infty} \tilde{\lambda}_i \tilde{f}_i(s) \tilde{f}_i(t), \]

with \( \tilde{\lambda}_i \) absolute summable and \( \tilde{f}_i \) distributional differentiable, by assumption.

On the other hand, \( K_{s,t} \) is a kernel and thus, by Mercer's theorem, admits eigenvectors \( \lambda > 0 \) and eigenfunctions \( f_i : \mathcal{T} \to \mathbb{R} \) such that \( \sum_i \lambda_i < \infty \) and

\[ K_{s,t} = \sum_{i=1}^{\infty} \lambda_i f_i(t) f_i(s). \]

In particular, since \( f_i \) is an eigenfunction of \( K_{s,t} \) it holds

\[
\begin{align*}
\lambda_k f_k(t) &= \int K_{s,t} f_k(s) ds \\
&= \int \sum_{i=1}^{\infty} \tilde{\lambda}_i \tilde{f}_i(t) \tilde{f}_i(s) f_k(s) ds \\
&= \sum_{i=1}^{\infty} \tilde{\lambda}_i \int \tilde{f}_i(s) f_k(s) ds \cdot \tilde{f}_i(t),
\end{align*}
\]
where \(!\) holds, since \(\hat{\lambda}_i\) is absolutely summable. Now, by Hölder’s inequality it follows that \(\int \hat{f}_i(s)f_k(s)ds \leq \int \hat{f}_i(s)^2ds \cdot \int f_k(s)^2ds \leq C\), thus \(\hat{\lambda}_i \int \hat{f}_i(s)f_k(s)ds\) is absolutely summable, too. As a consequence, \(f_k\) has to be distributional differentiable as well, since it can be written as the weak limite, of bounded, differentiable functions. The first statement now follows using the same arguments as in the proof of Lemma 1 by Lemma 2.

The second part follows from the fact that \(K_{s,t}\) is locally constant with jumps along \(t_i\). Thus, \(f_k\) is locally constant with jumps at \(t_i\) and therefore admits the distributional derivative

\[
\sqrt{\lambda_k}f_k(t) = \sum_{i=1}^{n} s_{i,j} \delta_{t_i}
\]

and it holds \((W \ast \delta_t) = W(\cdot - t)\). The sum property follows by choosing \(w\) with sufficiently small support, observing that \(s_{i,j}\) does not depend on \(w\), and using [16, Corollary 1].

**B Proofs for Section 4 (Spectral Drift Detection Method)**

**Theorem 2.** The first part follows directly from the definitions. The second part follows by the observation that \(\int f_k(t)f_l(t)dt = v_k^Tv_l\).

**C Proofs for Section 5 (Experiments)**

In addition to the results already presented in Section 5 we will provide additional analysis here. Table 2 summarizes the median number of alerts and thus provides an upper bound on the specificity of the methods. Table 3 documents the median delay between a drift event and the closest detected drift, i.e. if choosing this as max delay, 50% of all events are found. Similar information is provided in Fig. 4, which shows the number of detected drifts vs. maximal delay. Table 4 and Table 5 show the \(\beta\)-score (\(\beta = 0.5\)) for all split length with varying choices of maximal delay.
Table 2: Results over 150 runs. Table shows median number of alarms.

| DS               | Method      | AB  | ABA | ABC |
|------------------|-------------|-----|-----|-----|
|                  |             | -  | 25  | 50  | 75  | 100 |
| Electricity      |             |    | 2   | 3   | 3   | 3   | 3   | 3   |
| Market Prices    | D3          | 2  | 2   | 3   | 3   | 3   | 3   | 3   | 3   |
|                  | HDDDM       | 1  | 0   | 0   | 0   | 0   | 1   | 1   | 1   |
|                  | KSDD        | 12 | 11  | 11  | 11  | 12  | 12  | 12  | 12  |
|                  | MMDDDM      | 12 | 7   | 9   | 13  | 13  | 16  | 16  | 17  |
|                  | SDDM(MT)    | 1  | 0   | 2   | 2   | 2   | 2   | 2   | 2   |
|                  | SDDM(RBF)   | 1  | 0   | 0   | 0   | 0   | 2   | 2   | 2   |
|                  | ShapeDDM    | 1  | 0   | 0   | 0   | 1   | 1   | 1   | 1   |
| Forest           |             |    | 3   | 2   | 3   | 3   | 3   | 3   | 3   |
| Covertype        | D3          | 3  | 2   | 3   | 3   | 3   | 3   | 3   | 3   |
|                  | HDDDM       | 2  | 1   | 1   | 1   | 1   | 1   | 1   | 1   |
|                  | KSDD        | 31 | 32  | 35  | 36  | 36  | 32  | 31  | 30  |
|                  | MMDDDM      | 12 | 14  | 11  | 11  | 12  | 15  | 16  | 17  |
|                  | SDDM(MT)    | 1  | 3   | 2   | 3   | 3   | 2   | 2   | 2   |
|                  | SDDM(RBF)   | 0  | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
|                  | ShapeDDM    | 1  | 0   | 1   | 1   | 1   | 1   | 1   | 1   |
| Random           |             |    | 3   | 2   | 2   | 3   | 3   | 3   | 3   |
| RBF             | D3          | 3  | 2   | 2   | 3   | 3   | 3   | 3   | 3   |
|                  | HDDDM       | 2  | 0   | 0   | 0   | 1   | 2   | 1   | 1   |
|                  | KSDD        | 16 | 17  | 17  | 17  | 17  | 17  | 17  | 17  |
|                  | MMDDDM      | 9  | 11  | 15  | 16  | 16  | 10  | 13  | 16  |
|                  | SDDM(MT)    | 1  | 0   | 2   | 2   | 1   | 2   | 2   | 2   |
|                  | SDDM(RBF)   | 0  | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
|                  | ShapeDDM    | 1  | 0   | 1   | 1   | 1   | 1   | 1   | 1   |
| Rotating         |             |    | 1   | 1   | 1   | 1   | 1   | 2   | 2   |
| Hyperplane       | D3          | 1  | 1   | 1   | 1   | 1   | 2   | 2   | 2   |
|                  | HDDDM       | 1  | 1   | 1   | 1   | 1   | 2   | 1   | 1   |
|                  | KSDD        | 7  | 7   | 7   | 7   | 7   | 7   | 7   |
|                  | MMDDDM      | 12 | 7   | 5   | 6   | 8   | 13  | 14  | 17  |
|                  | SDDM(MT)    | 0  | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
|                  | SDDM(RBF)   | 2  | 0   | 0   | 0   | 2   | 2   | 2   | 2   |
|                  | ShapeDDM    | 1  | 0   | 0   | 0   | 0   | 1   | 1   | 1   |
| STAGGER          |             |    | 2   | 3   | 3   | 3   | 3   | 3   | 2   |
| Nebraska         |             |    | 2   | 3   | 3   | 3   | 3   | 3   | 3   |
| Weather          | D3          | 2  | 2   | 3   | 3   | 3   | 3   | 3   | 3   |
|                  | HDDDM       | 2  | 1   | 1   | 1   | 1   | 2   | 1   | 1   |
|                  | KSDD        | 3  | 3   | 3   | 3   | 4   | 3   | 3   | 4   |
|                  | MMDDDM      | 11 | 8   | 16  | 18  | 21  | 10  | 18  | 16  |
|                  | SDDM(MT)    | 1  | 1   | 2   | 2   | 2   | 2   | 2   | 2   |
|                  | SDDM(RBF)   | 1  | 0   | 0   | 2   | 1   | 2   | 2   | 2   |
|                  | ShapeDDM    | 1  | 0   | 1   | 1   | 2   | 1   | 1   | 1   |
|                  |             |    | 2   | 2   | 3   | 3   | 3   | 3   | 3   |
|                  | Nebraska     |    | 2   | 2   | 3   | 3   | 3   | 3   | 3   |
|                  | Weather       |    | 2   | 2   | 3   | 3   | 3   | 3   | 3   |

22
Table 3: Results over 150 runs. Table shows median distance between drift event and closes alarm.

| DS Method | Electricity Market Prices | Forest Covertype | Random RBF | Rotating Hyperplane | STAGGER | Nebraska Weather |
|-----------|---------------------------|------------------|------------|----------------------|---------|-------------------|
| D3        | 67 85 66 53 50 16 27 37 50 | 169 176 200 197 212 135 171 78 85 | 30 23 28 37 50 20 29 37 50 | 197 76 34 38 50 16 28 37 50 | 13 278 88 76 73 27 42 50 64 |
| HDDDM     | 28 263 264 230 397 31 47 56 67 | 321 373 344 385 286 260 348 342 334 | 36 157 102 311 72 66 75 80 | 311 338 308 235 253 178 74 71 80 | |
| KSDD      | 21 23 25 37 37 12 16 11 8   | 83 50 73 64 60 58 74 53 55 | 16 36 41 39 48 22 31 38 50 | 15 12 18 14 11 10 12 7 5 | |
| MMDDDM    | 10 134 29 26 26 6 5 6 6    | 2 13 14 10 11 9 7 3 3 | 4 12 8 8 6 10 11 7 6 | 57 88 21 17 14 4 6 8 5 | |
| SDDM(MT)  | 2 3 3 2 3 1 1 0 1         | 2 12 2 1 2 12 3 1 2 | 1 1 1 0 1 2 2 1 2 | 4 13 3 3 4 4 2 1 2 | |
| SDDM(RBF) | 5 203 57 37 51 3 3 3 3    | 3 - - - - - - 31 37 50 | 2 0 1 0 1 12 2 1 2 | 9 1 25 26 9 5 2 1 2 | |
| ShapeDDM  | 9 210 144 85 75 15 25 37 50 | 10 134 29 26 26 6 5 6 6 | 2 0 1 0 1 12 2 1 2 | 4 26 28 450 301 292 12 25 37 50 | |
Figure 4: Rate of detected drifts (y-axis) for different choices of max delay (x-axis).
Table 4: Results over 150 runs. Table shows mean $\beta$-score ($\beta = 0.5$) and standard deviation. Numbers below stream types are B-length and maximal delay.

| DS  | Method      | Electricity Prices | Forest Covertype | Random RBF |
|-----|-------------|-------------------|-----------------|------------|
|     |             | AB                | ABA             | AB         |
|     |             | -50/25            | 25/12           | 50/25      |
|     | D3          | 0.23 ±0.30        | 0.09 ±0.16      | 0.13 ±0.17 |
|     | HDDDM       | 0.65 ±0.44        | 0.00 ±0.00      | 0.01 ±0.07 |
|     | KSDD        | 0.12 ±0.08        | 0.07 ±0.08      | 0.14 ±10   |
|     | MMDDDM      | 0.17 ±0.09        | 0.02 ±0.06      | 0.14 ±10   |
|     | SDDM(MT)    | **0.89±0.25**     | **0.20±0.36**   | **0.65±0.40** |
|     | SDDM(RBF)   | 0.55 ±0.45        | 0.00 ±0.00      | 0.01 ±0.07 |
|     | ShapeDDM    | **0.93±0.18**     | 0.00 ±0.04      | 0.03 ±11   |
|     |             | 0.46 ±0.27        | 0.21 ±0.18      | 0.26 ±15   |
|     | D3          | 0.57 ±0.38        | 0.03 ±0.11      | 0.12 ±21   |
|     | HDDDM       | 0.07 ±0.01        | 0.07 ±0.03      | 0.09 ±03   |
|     | KSDD        | 0.17 ±0.09        | 0.11 ±0.12      | 0.29 ±15   |
|     | MMDDDM      | 0.92 ±0.15        | **0.69±0.32**   | **0.89±0.13** |
|     | SDDM(MT)    | 0.03 ±0.16        | 0.00 ±0.00      | 0.00 ±00   |
|     | SDDM(RBF)   | **0.98±0.07**     | 0.02 ±0.10      | 0.22 ±25   |
|     | ShapeDDM    | 0.31 ±0.30        | 0.12 ±0.17      | 0.18 ±19   |
|     | D3          | 0.35 ±0.36        | 0.01 ±0.08      | 0.03 ±11   |
|     | HDDDM       | 0.11 ±0.04        | 0.08 ±07       | 0.13 ±07   |
|     | KSDD        | 0.21 ±0.10        | 0.10 ±0.13      | 0.22 ±09   |
|     | MMDDDM      | **0.93±0.16**     | 0.20 ±0.35      | **0.78±0.31** |
|     | SDDM(MT)    | 0.06 ±0.23        | 0.00 ±0.00      | 0.01 ±08   |
|     | SDDM(RBF)   | **0.97±0.10**     | 0.01 ±0.08      | 0.14 ±22   |
|     | ShapeDDM    |                      |                     |            |
| Rotating Hyperplane | D3          | HDDDM     | KSDD      | MMDDDM     | SDDM(MT)   | SDDM(RBF)  | ShapeDDM   |
|---------------------|-------------|-----------|-----------|------------|------------|------------|------------|
|                     | 0.14 ± 0.30 | 0.03 ± 0.18 | 0.08 ± 0.13 | 0.17 ± 0.09 | 0.25 ± 0.42 | 0.78 ± 0.19 | 0.91 ± 0.15 |
|                     | 0.07 ± 0.17 | 0.02 ± 0.09 | 0.07 ± 0.11 | 0.07 ± 0.14 | 0.00 ± 0.09 | 0.00 ± 0.00 | 0.00 ± 000 |
|                     | 0.07 ± 0.16 | 0.04 ± 0.13 | 0.11 ± 0.12 | 0.08 ± 0.12 | 0.00 ± 0.03 | 0.00 ± 0.00 | 0.70 ± 0.29 |
|                     | 0.11 ± 0.18 | 0.00 ± 0.04 | 0.00 ± 0.00 | 0.17 ± 0.08 | 0.18 ± 0.23 | 0.00 ± 0.00 | 0.75 ± 0.25 |
|                     | 0.19 ± 0.21 | 0.03 ± 0.11 | 0.13 ± 0.13 | 0.21 ± 0.10 | 0.20 ± 0.25 | 0.00 ± 0.00 | 0.80 ± 0.21 |
|                     | 0.19 ± 0.22 | 0.03 ± 0.11 | 0.17 ± 0.15 | 0.22 ± 0.08 | 0.23 ± 0.29 | 0.00 ± 0.00 | 0.80 ± 0.22 |

| STAGGER            | D3          | HDDDM     | KSDD      | MMDDDM     | SDDM(MT)   | SDDM(RBF)  | ShapeDDM   |
|---------------------|-------------|-----------|-----------|------------|------------|------------|------------|
|                     | 0.34 ± 0.31 | 0.41 ± 0.36 | 0.31 ± 0.27 | 0.18 ± 0.07 | 0.88 ± 0.30 | 0.71 ± 0.33 | 0.94 ± 0.12 |
|                     | 0.36 ± 0.23 | 0.15 ± 0.23 | 0.17 ± 0.17 | 0.24 ± 0.13 | 0.66 ± 0.43 | 0.24 ± 0.33 | 0.16 ± 0.22 |
|                     | 0.42 ± 0.26 | 0.12 ± 0.21 | 0.27 ± 0.23 | 0.23 ± 0.09 | 0.68 ± 0.46 | 0.50 ± 0.38 | 0.09 ± 0.18 |
|                     | 0.18 ± 0.17 | 0.06 ± 0.15 | 0.16 ± 0.15 | 0.18 ± 0.06 | 0.68 ± 0.33 | 0.49 ± 0.37 | 0.35 ± 0.17 |
|                     | 0.28 ± 0.18 | 0.01 ± 0.08 | 0.11 ± 0.17 | 0.19 ± 0.10 | 0.72 ± 0.38 | 0.31 ± 0.33 | 0.24 ± 0.25 |
|                     | 0.22 ± 0.21 | 0.04 ± 0.04 | 0.02 ± 0.10 | 0.13 ± 0.05 | 0.82 ± 0.30 | 0.42 ± 0.30 | 0.28 ± 0.24 |
|                     | 0.03 ± 0.11 | 0.07 ± 0.06 | 0.11 ± 0.05 | 0.14 ± 0.04 | 0.94 ± 0.12 | 0.50 ± 0.32 | 0.27 ± 0.22 |
|                     | 0.04 ± 0.13 | 0.06 ± 0.12 | 0.12 ± 0.12 | 0.12 ± 0.04 | 0.87 ± 0.14 | 0.30 ± 0.20 | 0.33 ± 0.20 |

| Nebraska Weather   | D3          | HDDDM     | KSDD      | MMDDDM     | SDDM(MT)   | SDDM(RBF)  | ShapeDDM   |
|---------------------|-------------|-----------|-----------|------------|------------|------------|------------|
|                     | 0.60 ± 0.22 | 0.01 ± 0.08 | 0.11 ± 0.17 | 0.08 ± 0.03 | 0.82 ± 0.30 | 0.09 ± 0.27 | 0.66 ± 0.44 |
|                     | 0.12 ± 0.17 | 0.02 ± 0.10 | 0.12 ± 0.12 | 0.07 ± 0.06 | 0.49 ± 0.43 | 0.03 ± 0.16 | 0.01 ± 0.07 |
|                     | 0.18 ± 0.17 | 0.03 ± 0.11 | 0.20 ± 0.13 | 0.13 ± 0.05 | 0.63 ± 0.42 | 0.10 ± 0.26 | 0.12 ± 0.02 |
|                     | 0.24 ± 0.21 | 0.00 ± 0.00 | 0.18 ± 0.11 | 0.04 ± 0.14 | 0.64 ± 0.42 | 0.33 ± 0.39 | 0.03 ± 0.12 |
|                     | 0.32 ± 0.25 | 0.09 ± 0.19 | 0.24 ± 0.10 | 0.12 ± 0.04 | 0.91 ± 0.31 | 0.40 ± 0.43 | 0.19 ± 0.21 |
|                     | 0.45 ± 0.25 | 0.17 ± 0.23 | 0.25 ± 0.09 | 0.13 ± 0.04 | 0.91 ± 0.13 | 0.49 ± 0.39 | 0.18 ± 0.24 |
|                     | 0.45 ± 0.25 | 0.17 ± 0.23 | 0.25 ± 0.09 | 0.13 ± 0.04 | 0.91 ± 0.13 | 0.49 ± 0.39 | 0.18 ± 0.24 |
Table 5: Results over 150 runs. Table shows mean $\beta$-score ($\beta = 0.5$) and standard deviation. Numbers below stream types are B-length. Maximal delay is 12.

| DS               | Method    | AB   | ABA   | ABC   |
|------------------|-----------|------|-------|-------|
|                  |           | -12  | 25/12 | 50/12 | 75/12 | 100/12 | 25/12 | 50/12 | 75/12 | 100/12 |
| Electricity      | D3        | 0.11±0.25 | 0.09±0.16 | 0.10±0.16 | 0.12±0.17 | 0.14±0.20 | 0.29±0.15 | 0.22±0.18 | 0.20±0.17 | 0.30±0.28 |
|                  | HDDDM     | 0.18±0.38 | 0.00±0.00 | 0.01±0.06 | 0.00±0.04 | 0.01±0.06 | 0.18±0.23 | 0.11±0.20 | 0.10±0.20 | 0.10±0.20 |
|                  | KSDD      | 0.06±0.09 | 0.07±0.08 | 0.08±0.08 | 0.10±0.09 | 0.09±0.10 | 0.18±0.09 | 0.16±0.10 | 0.17±0.10 | 0.19±0.11 |
|                  | MMDDDM    | 0.09±0.10 | 0.02±0.06 | 0.06±0.10 | 0.05±0.08 | 0.05±0.08 | 0.17±0.08 | 0.19±0.11 | 0.18±0.10 | 0.14±0.09 |
| Market Prices    | SDDM(MT)  | **0.86±0.30** | **0.20±0.36** | **0.59±0.41** | **0.71±0.36** | **0.78±0.29** | **0.86±0.22** | **0.95±0.11** | **0.96±0.09** | **0.97±0.09** |
|                  | SDDM(RBF) | 0.39±0.45 | 0.00±0.00 | 0.01±0.05 | 0.03±0.15 | 0.02±0.11 | 0.64±0.36 | 0.62±0.36 | 0.70±0.35 | 0.66±0.37 |
|                  | ShapeDDM  | 0.59±0.47 | 0.00±0.04 | 0.02±0.09 | 0.02±0.10 | 0.04±0.14 | 0.37±0.21 | 0.29±0.24 | 0.26±0.25 | 0.22±0.24 |
| Forest Covertype | D3        | 0.40±0.29 | 0.21±0.18 | 0.22±0.17 | 0.24±0.15 | 0.56±0.27 | 0.29±0.15 | 0.26±0.16 | 0.28±0.15 | 0.41±0.30 |
|                  | HDDDM     | 0.16±0.31 | 0.03±0.11 | 0.04±0.13 | 0.05±0.15 | 0.03±0.12 | 0.17±0.21 | 0.14±0.21 | 0.10±0.19 | 0.08±0.17 |
|                  | KSDD      | 0.05±0.03 | 0.07±0.03 | 0.07±0.03 | 0.09±0.03 | 0.10±0.02 | 0.09±0.04 | 0.08±0.04 | 0.09±0.04 | 0.09±0.04 |
|                  | MMDDDM    | 0.15±0.09 | 0.11±0.12 | 0.10±0.11 | 0.19±0.09 | 0.19±0.10 | 0.19±0.13 | 0.16±0.09 | 0.20±0.08 | 0.22±0.10 |
|                  | SDDM(MT)  | **0.90±0.19** | **0.69±0.32** | **0.89±0.13** | **0.86±0.14** | **0.86±0.15** | **0.66±0.24** | **0.85±0.22** | **0.89±0.17** | **0.89±0.18** |
|                  | SDDM(RBF) | 0.03±0.16 | 0.00±0.00 | 0.00±0.00 | 0.00±0.00 | 0.00±0.00 | 0.00±0.00 | 0.00±0.00 | 0.02±0.12 | 0.01±0.07 |
|                  | ShapeDDM  | **0.92±0.25** | 0.02±0.10 | 0.15±0.23 | 0.22±0.25 | 0.30±0.24 | 0.47±0.11 | 0.07±0.18 | 0.03±0.11 | 0.02±0.09 |
| Random RBF       | D3        | 0.22±0.30 | 0.12±0.17 | 0.13±0.18 | 0.20±0.17 | 0.23±0.26 | 0.15±0.16 | 0.24±0.18 | 0.21±0.18 | 0.32±0.29 |
|                  | HDDDM     | 0.07±0.21 | 0.01±0.08 | 0.01±0.07 | 0.01±0.07 | 0.04±0.13 | 0.11±0.17 | 0.02±0.10 | 0.03±0.11 | 0.03±0.12 |
|                  | KSDD      | 0.07±0.06 | 0.08±0.07 | 0.09±0.07 | 0.12±0.08 | 0.12±0.08 | 0.12±0.08 | 0.12±0.08 | 0.11±0.07 | 0.13±0.07 |
|                  | MMDDDM    | 0.19±0.11 | 0.10±0.13 | 0.15±0.09 | 0.19±0.11 | 0.23±0.11 | 0.23±0.12 | 0.18±0.08 | 0.21±0.10 | 0.20±0.09 |
|                  | SDDM(MT)  | **0.92±0.18** | **0.20±0.35** | **0.74±0.32** | **0.86±0.21** | **0.89±0.19** | **0.56±0.22** | **0.80±0.27** | **0.86±0.22** | **0.88±0.20** |
|                  | SDDM(RBF) | 0.06±0.22 | 0.00±0.00 | 0.01±0.08 | 0.06±0.21 | 0.09±0.26 | 0.01±0.07 | 0.12±0.25 | 0.15±0.28 | 0.20±0.34 |
|                  | ShapeDDM  | **0.93±0.21** | 0.01±0.08 | 0.08±0.18 | 0.16±0.23 | 0.27±0.24 | 0.48±0.09 | 0.17±0.23 | 0.13±0.22 | 0.09±0.19 |
|          | D3       | HDDDM    | KSDD     | MMDDDM   | SDDM(MT) | SDDM(RBF) | ShapeDDM |
|----------|----------|----------|----------|----------|----------|-----------|----------|
| Rotating |          |          |          |          |          |           |          |
| Hyperplane| 0.06 ± 0.19 | 0.01 ± 0.12 | 0.02 ± 0.07 | 0.17 ± 0.09 | 0.24 ± 0.41 | 0.75 ± 0.25 | 0.91 ± 0.17 |
| STAGGER  |          |          |          |          |          |           |          |
| 0.19 ± 0.29 | 0.14 ± 0.28 | 0.22 ± 0.28 | 0.16 ± 0.09 | 0.87 ± 0.32 | 0.71 ± 0.34 | 0.92 ± 0.20 |
|           | 0.17 ± 0.18 | 0.01 ± 0.08 | 0.14 ± 0.17 | 0.16 ± 0.15 | 0.46 ± 0.47 | 0.11 ± 0.28 | 0.06 ± 0.17 |
|           | 0.16 ± 0.19 | 0.02 ± 0.08 | 0.12 ± 0.16 | 0.16 ± 0.11 | 0.64 ± 0.43 | 0.21 ± 0.29 | 0.09 ± 0.18 |
|           | 0.20 ± 0.17 | 0.07 ± 0.17 | 0.18 ± 0.20 | 0.15 ± 0.09 | 0.72 ± 0.42 | 0.45 ± 0.39 | 0.04 ± 0.14 |
|           | 0.27 ± 0.28 | 0.04 ± 0.13 | 0.22 ± 0.24 | 0.16 ± 0.17 | 0.65 ± 0.46 | 0.46 ± 0.38 | 0.27 ± 0.21 |
|           | 0.18 ± 0.17 | 0.06 ± 0.14 | 0.17 ± 0.20 | 0.17 ± 0.19 | 0.68 ± 0.33 | 0.31 ± 0.33 | 0.24 ± 0.25 |
|           | 0.21 ± 0.20 | 0.09 ± 0.17 | 0.24 ± 0.25 | 0.19 ± 0.21 | 0.69 ± 0.38 | 0.36 ± 0.28 | 0.16 ± 0.23 |
|           | 0.17 ± 0.17 | 0.05 ± 0.14 | 0.22 ± 0.23 | 0.22 ± 0.19 | 0.79 ± 0.36 | 0.48 ± 0.33 | 0.25 ± 0.21 |
| Nebraska |          |          |          |          |          |           |          |
| Weather   | 0.03 ± 0.14 | 0.01 ± 0.08 | 0.04 ± 0.07 | 0.02 ± 0.07 | 0.69 ± 0.41 | 0.06 ± 0.22 | 0.33 ± 0.46 |
|           | 0.12 ± 0.17 | 0.00 ± 0.04 | 0.01 ± 0.07 | 0.06 ± 0.12 | 0.14 ± 0.30 | 0.03 ± 0.16 | 0.01 ± 0.07 |
|           | 0.14 ± 0.18 | 0.01 ± 0.07 | 0.01 ± 0.07 | 0.06 ± 0.06 | 0.47 ± 0.43 | 0.10 ± 0.25 | 0.01 ± 0.08 |
|           | 0.21 ± 0.26 | 0.01 ± 0.08 | 0.01 ± 0.07 | 0.08 ± 0.06 | 0.57 ± 0.42 | 0.29 ± 0.37 | 0.06 ± 0.15 |
|           | 0.23 ± 0.17 | 0.00 ± 0.00 | 0.01 ± 0.07 | 0.09 ± 0.06 | 0.54 ± 0.41 | 0.33 ± 0.39 | 0.07 ± 0.17 |
|           | 0.18 ± 0.17 | 0.05 ± 0.15 | 0.03 ± 0.12 | 0.10 ± 0.04 | 0.66 ± 0.31 | 0.49 ± 0.39 | 0.17 ± 0.23 |
|           | 0.21 ± 0.17 | 0.03 ± 0.12 | 0.03 ± 0.12 | 0.09 ± 0.04 | 0.88 ± 0.18 | 0.78 ± 0.23 | 0.06 ± 0.16 |
|           | 0.22 ± 0.28 | 0.06 ± 0.14 | 0.05 ± 0.11 | 0.11 ± 0.04 | 0.87 ± 0.20 | 0.85 ± 0.23 | 0.06 ± 0.16 |