Kernel methods for vessel trajectories

de Vries, G.K.D.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
In this chapter we study similarity measures for vessel trajectories that are based on taking the integral over time of the distance between two trajectories. For these similarities we define positive semi-definite kernel variants in an approach similar to Chapter 4. We test these kernels in the tasks of clustering, classification and outlier detection. These experiments indicate that these kernel variants work, but they are outperformed by the alignment methods from the previous chapter. As in the previous chapter the soft-max kernel variants perform well in outlier detection.

5.1 INTRODUCTION

The alignment measures used in clustering, classification and outlier detection in Chapter 4 are one type of similarity that can be used for these tasks. Another solution, coming from the field of computational geometry is to express the similarity between trajectories as an integral of the distance between two trajectories over time. To deal with the temporal length and distance traveled variations in trajectories, this similarity has the option of allowing time shifts and subtrajectories.

As in the chapter about alignments, we study this similarity in the kernel framework. Therefore we define, for each combination of allowing time shift and/or subtrajectories, positive semi-definite kernel variants of this similarity measure. To the best of our knowledge these positive semi-definite kernels have not been studied in the field of spatio-temporal data-mining. However, some of the kernels that we define are equal to kernels already considered in the time-series literature. As in Chapter 4 we also consider direct variants of the defined similarity measure, which are not positive semi-definite.

We use the defined kernels in the same experiments as Chapter 4. The goal of these experiments is to discover which of the variants of the similarity measure performs best and how the performance compares to the alignment kernels. It turns out that the alignment kernels outperform the integral based kernels.

In the rest of this chapter we discuss the following. Section 5.2 defines a general version of trajectory similarity based on the integral over time. We then give more specific versions of this similarity and corresponding positive semi-definite kernels. The experiments with clustering, classification and outlier detection, using these ker-
5.2 Integral based trajectory similarity

Buchin et al. (2009) give algorithms to compute various cases of Definition 5.2.1, a form of integral over time based trajectory similarity. A more specific version of this similarity appeared in earlier literature, such as the work by Nanni and Pedreschi (2006). There exist other definitions for shape matching that do not explicitly consider time that can also be applied to trajectories, but we do not consider those here.

In Definition 5.2.1, trajectories are continuous functions $T : \mathbb{R} \rightarrow \mathbb{R} \times \mathbb{R}$, from time to position, i.e. $T(t) = (x,y)$. We use bold face capital letters ($T$) to distinguish trajectories as continuous functions from trajectories as sequences ($T$) as in Definition 2.2.1. Furthermore, for two trajectories $S, T$, let $d$ be the Euclidean distance function:

$$d(S(t), T(t)) = \| (S(t) - T(t)) \| .$$  \hspace{1cm} (5.1)

**Definition 5.2.1.** Let $S$ and $T$ be two trajectories. And let $\tau \leq \min(t_s, t_T - t_{shift})$, where $t_s$ is the maximum $t$ for which $S(t)$ exists and $t_T$ is the maximum $t$ for which $T(t)$ exists. Furthermore, $\tau \geq t_{min} > 0$. Then, we define trajectory similarity $D_{sub\_shift}(S, T)$ between $S$ and $T$ as

$$D_{sub\_shift}(S, T) = \min_{\tau, t_{shift}} \frac{\int_0^\tau d(S(t), T(t + t_{shift}))dt}{\tau} .$$

Figure 11 illustrates this similarity function. The variable $\tau$ represents the temporal length over which the integral of $d$ over time is computed. With $t_{shift}$ the trajectory $T$ can be temporally shifted with respect to the trajectory $S$. The similarity value is the value of the integral over time divided by $\tau$ for the $\tau$ and $t_{shift}$ that minimize this integral divided by $\tau$. Furthermore $\tau$ is constrained such that it is larger than some minimal subtrajectory length $t_{min}$.

In this rest of this section we will first consider the more specific cases of this similarity. We consider the version that does not allow for...
subtrajectories, i.e. where $\tau$ is fixed, both with and without time shift $t_{\text{shift}}$ in Section 5.2.1. Then we treat the versions with subtrajectories, i.e. variable $\tau$, in Section 5.2.2. For these four variants we introduce corresponding positive semi-definite kernels. Most of these kernels are similar to the soft-max kernels of Chapter 4. We also consider direct kernels of the four variants. A list of all the kernels introduced in this chapter is given in Table 10.

As mentioned in Chapter 2, we use the term ‘similarity’ generically, for similarities and distances alike. Hence we refer to the distance defined in Definition 5.2.1 as a similarity. Furthermore, we recall that we refer to both a kernel function and the kernel matrix based on that function as kernel. However, we use the small $k$ for functions and the capital $K$ for matrices. We use the concept of a mapping kernel (Shin and Kuboyama (2008)), given in Definition 2.1.2, to ensure PSD-ness of some of the kernels that we define. We could also do this directly via the ‘standard’ closure properties (Section 2.1.1.1). However, we think that the mapping kernel leads to more elegant definitions.
5.2.1 Full Trajectory Similarity Kernels

In this section we look at variants of Definition 5.2.1 that do not allow subtrajectories. We define positive semi-definite kernels corresponding to these similarities.

5.2.1.1 No Time Shift

First, we consider the case of Definition 5.2.1 where \( \tau = \min(t_S, t_T) \), i.e. we consider full trajectory similarity. We also do not allow for time shift, which reduces Definition 5.2.1 to Definition 5.2.2 below.

**Definition 5.2.2.** Let \( S \) and \( T \) be two trajectories. And let \( \tau = \min(t_S, t_T) \), where \( t_S \) is the maximum \( t \) for which \( S(t) \) exists and \( t_T \) is the maximum \( t \) for which \( T(t) \) exists. Then, we define trajectory similarity \( D_{\text{noshift}}(S, T) \) between \( S \) and \( T \) as

\[
D_{\text{noshift}}(S, T) = \frac{\int_0^\tau d(S(t), T(t)) \, dt}{\tau} .
\]

Thus, the integral over time is computed over the entire temporal length for which the two trajectories both exist. This similarity is the same as what time-series literature refers to as the Euclidean distance for time-series (Faloutsos et al. (1994)).

Trajectories are always sampled and therefore piecewise linear, which leads to the fact that \( d \) is a piecewise hyperbolic function. Each piece \( d^* \) of \( d \) is of the form:

\[
d^*(t) = \sqrt{At^2 + Bt + C} ,
\]

for some constants \( A, B, C \), which leads to a symbolically solvable integral for each piece. Note that the similarity function is a metric, if the time \( \tau \) is the same for every two trajectories in our set.

Before we turn the similarity measure in Definition 5.2.2 into a positive semi-definite kernel we recall the kernel closure properties from Section 2.1.1.1.1.

Furthermore, we know that the Gaussian kernel, with scaling parameter \( \beta > 0 \),

\[
\exp(-\beta d(S(t), T(t))^2)
\]

is positive semi-definite.

In the vein of the similarity in Definition 5.2.2 we define the following kernel \( k \), for two trajectories \( S, T \in \mathcal{U} \), where \( \mathcal{U} \) contains trajectories of the same length \( \tau \). Let the interval \([0, \tau]\) be divided into subintervals of equal width \( \Delta t \), and from each interval choose a point \( t_i \), then

\[
k^\beta(S, T) = \int_0^\tau \exp(-\beta d(S(t), T(t))^2) \, dt
\]

\[
= \lim_{n \to \infty} \sum_{i=1}^n \exp(-\beta d(S(t_i), T(t_i))^2) \Delta t .
\]
We can see that this is a proper kernel by the closure under sum and limit properties, since \( \tau \) is finite, and \( \exp(-\beta d(S(t), T(t))^2) \) is a kernel.

We can extend this kernel to trajectories of variable length. Suppose \( \mathcal{V} \) is a set of trajectories with variable temporal length. Let \( t_{\text{ends}} = 0, t_1, \ldots, t_m \) be the ordered list of the last time points for each \( T \in \mathcal{V} \), starting with 0. Let \( T(t_i, t_j) \) be the part of the trajectory \( T \) from \( t_i \) to \( t_j \). Now consider the following kernel \( k' \), with \( S, T \in \mathcal{V} \).

\[
  k'(S, T) = \sum_{n=1}^{m} k(S(t_{\text{ends}}(n), t_{\text{ends}}(n+1)), T(t_{\text{ends}}(n), t_{\text{ends}}(n+1))) ,
\]

where \( k(S(t_{\text{ends}}(n), t_{\text{ends}}(n+1)), T(t_{\text{ends}}(n), t_{\text{ends}}(n+1))) \) is 0 if either \( S \) or \( T \) does not exist for the interval \( t_{\text{ends}}(n), t_{\text{ends}}(n+1) \). Note that by zero extension \( k \) remains a valid kernel. Hence, by closure under sum, \( k' \) is a valid kernel.

In practice the integral defined in the above kernel is computationally intensive to compute compared to approximating the integral by sampling \( d(S(t), T(t)) \) at a fixed sample rate, especially when considering time shift, as we will do further on in the chapter.

From now on we will only consider the fixed sample rate case, instead of the somewhat more complex case of continuous trajectory functions. We do this for two reasons. Firstly, the following kernel definitions are clearer and easier to read for the fixed sample rate case. Secondly and more importantly, in practice we use an approximation by fixed sample rate to compute both the similarity functions \( D(S, T) \) and the kernels that we define.

In the rest of this chapter a fixed sample rate trajectory is a sequence of points \( (T = \langle x_1, y_1, t_1 \rangle, \langle x_2, y_2, t_2 \rangle, \ldots, \langle x_n, y_n, t_n \rangle) \), as in Definition 2.2.1, indicated with an italic font face \( (T) \). We recall the shorthand \( T(i) = \langle x_i, y_i, t_i \rangle \). Since the sample rate is fixed, \( d(S(i), T(i)) \) reduces to \( d((x_{S,i}, y_{S,i}), (x_{T,i}, y_{T,i})) \), with \( S(i) = \langle x_{S,i}, y_{S,i}, t_{S,i} \rangle \) and \( T(i) = \langle x_{T,i}, y_{T,i}, t_{T,i} \rangle \), because \( t_{S,i} = t_{T,i} \). I.e. in practice \( d \) is the same as in the continuous trajectory case.

Below we give the fixed sample rate version definition of kernel \( k' \).

**Definition 5.2.3.** Let \( \mathcal{T} \) be a set of fixed sample rate trajectories and \( \beta > 0 \). Then we define the kernel function \( k_{\text{noshift}} \) between two trajectories \( S, T \in \mathcal{T} \) as:

\[
  k_{\text{noshift}}^\beta(S, T) = \lim_{n \to \infty} \sum_{i=1}^{n} \exp(-\beta d(S(i), T(i))^2) ,
\]

where \( \exp(-\beta d(S(i), T(i))^2) \) is defined to be 0 if one of the trajectories is undefined for that \( i \).
This is finite, because trajectories have finite length. Like before, this is a valid kernel because of the closure properties under sum and limit.

Note that the kernel defined above allows us to compare a set of trajectories of different length. On the other hand, the original $D_{\text{noshift}}(S, T)$ is not a metric for a set of trajectories with different temporal length, and using the simple approach of directly putting $D$ in the Gaussian kernel (exp$^{-D(S, T)^2}$) does not yield a proper positive semi-definite kernel for trajectories of different length.

As mentioned above, we use the concept of a mapping kernel (Shin and Kuboyama (2008)), given in Definition 2.1.2, further on in this chapter to show that the defined kernels are PSD. We illustrate the use of the mapping kernel by showing that the above kernel is also a mapping kernel.

We define $\mathcal{J}'$ as the set of all $(x, y)$ points in the trajectories $\mathcal{J}$, and $\mathcal{J}'_T$ as the subset of $\mathcal{J}'$ that contains only the points of the trajectory $T$. Furthermore, we define the map $M_{\text{noshift}}(S, T)$ below:

$$M_{\text{noshift}}(S, T) = \{(S(i), T(i)) \mid 0 \leq i \leq \min(|S|, |T|)\}. \quad (5.7)$$

Each point in $S$ is mapped to the point in $T$ with the same index, if it exists. It is easy to see that this map is transitive. Consider any three trajectories $S, T, U \in \mathcal{J}$ and the three maps $M_{\text{noshift}}(S, T), M_{\text{noshift}}(T, U)$ and $M_{\text{noshift}}(S, U)$. If, for any $i$, $(S(i), T(i)) \in M_{\text{noshift}}(S, T)$ and $(T(i), U(i)) \in M_{\text{noshift}}(T, U)$, then $0 \leq i \leq |S|$ and $i \leq |U|$. Thus $(S(i), U(i)) \in M_{\text{noshift}}(S, U)$ by definition of the map.

With this map, we can give an alternative version of Definition 5.2.3:

$$k^\beta_{\text{noshift}}(S, T) = \sum_{(S(i), T(i)) \in M_{\text{noshift}}(S, T)} \exp(-\beta d(S(i), T(i))^2) \quad (5.8)$$

5.2.1.2 Time Shift

Now, we consider the variant of Definition 5.2.1 that includes time shift, but not subtrajectories. This version is given below in Definition 5.2.4.

**Definition 5.2.4.** Let $S$ and $T$ be two trajectories. And let $\tau = \min(t_S, t_T - t_{\text{shift}})$, where $t_S$ is the maximum $t$ for which $S(t)$ exists and $t_T$ is the maximum $t$ for which $T(t)$ exists. Furthermore, $\tau \geq t_{\text{min}} > 0$. Then, we define trajectory similarity $D_{\text{shift}}(S, T)$ between $S$ and $T$ as

$$D_{\text{shift}}(S, T) = \min_{t_{\text{shift}}} \frac{\int_{\tau}^{\tau} d(S(t), T(t + t_{\text{shift}}))\,dt}{\tau}. \quad (5.9)$$

Essentially this measure computes the integral over time from $D_{\text{noshift}}$ for the value of $t_{\text{shift}}$, i.e. the time shift, that minimizes this function. In other words, this similarity tries to optimally align $S$ and $T$ temporarily, such that the integral is smallest.
It makes sense to constrain the maximum allowed time shift $t_{\text{shift}}$ to be larger than some $t_{\text{min}}$. For example, a shift of $t_S$ or $t_T$ only compares the first and last point of two trajectories. This would make two trajectories with the same path, but in opposite direction, completely equal to each other.

To introduce time shift into the kernel of Definition 5.2.3, we treat time shift as a shift in a number of trajectory points ($i_{\text{shift}}$). Instead of comparing the points $S(i)$ and $T(i)$, we compare $S(i)$ with $T(i + i_{\text{shift}})$. Thus, naively incorporating time shift, in a similar fashion as in Definition 5.2.4, into Definition 5.2.3 leads to the following kernel:

$$k^\beta(S, T) = \min_{i_{\text{shift}}} \left( \lim_{n \to \infty} \sum_{i=1}^{n} \exp(-\beta d(S(i), T(i + i_{\text{shift}}))^2) \right) . \quad (5.10)$$

The combination of the optimal value of $i_{\text{shift}}$ for a pair of trajectories $S$ and $T$ and the optimal $i_{\text{shift}}$ for a pair $T$, $U$, does not necessarily give the optimal $i_{\text{shift}}$ for the pair of trajectories $S$ and $U$. This results in the fact that a map $M(S, T)$ similar to the one above is not transitive for this kernel, because of $i_{\text{shift}}$. The above kernel is not positive semi-definite. We can create a positive semi-definite kernel that takes into account time shift by defining a map $M(S, T)$ that is transitive and that represents this time shift in some fashion.

One obvious solution is to sum over all possible $i_{\text{shift}}$ between two trajectories $S$ and $T$, like the soft-max kernels in Chapter 4. However, this does not result in a PSD kernel. Somewhat more complex maps are required to solve this problem. In the rest of this section we will define two different kernels and corresponding maps that give PSD solutions which take into account time shift.

Suppose that all trajectories in $T$ are sampled with sample rate $\phi$. Let

$$\text{shift}(T, i_{\text{shift}}) = \langle x_1, y_1, t_1 - \frac{i_{\text{shift}}}{\phi}, \ldots, \langle x_{|T|}, y_{|T|}, t_{|T|} - \frac{i_{\text{shift}}}{\phi} \rangle ,$$

with $T(i) = \langle x_i, y_i, t_i \rangle . \quad (5.11)$

Then, $\text{shift}(T, i_{\text{shift}})$ defines a time shifted version of $T$ over $\frac{i_{\text{shift}}}{\phi}$. We define the set of all time shifts $\text{shifts}(T)$ for one trajectory $T$ as:

$$\text{shifts}(T) = \{ \text{shift}(T, i_{\text{shift}}) \mid 0 \leq i_{\text{shift}} \leq |T| \} . \quad (5.12)$$

In terms of a mapping kernel, if $T$ is a set of trajectories, then $T' = \bigcup_{T \in T} \text{shifts}(T)$, and hence $T'_T = \text{shifts}(T)$. As a map we take

$$M_{\text{sumshift}}(S, T) = \text{shifts}(S) \times \text{shifts}(T) . \quad (5.13)$$

This map is trivially transitive, since it consists of the cross-product.

To construct a mapping kernel with this map, we need a kernel on $T'$. This kernel is a slightly altered version of $k_{\text{noshift}}$. We need to change the kernel, since trajectories in $T'$ do not necessarily have
t = 0 for the first element, hence we cannot directly compare \( S(i) \) to \( T(i) \), since the t’s might not be equal. We give this altered version based on the mapping kernel variant of \( k_{\text{noshift}} \). We need to redefine the map \( M'_{\text{noshift}}(S, T) \) from Equation 5.7 to \( M'_{\text{noshift}}(S, T) \) such that each element in \( S \) is mapped to the element in \( T \) with the same time \( t \).

\[
M'_{\text{noshift}}(S, T) = \{(S(i)), T(j)) \mid t_i = t_j, S(i) = (x_i, y_i, t_i), T(j) = (x_j, y_j, t_j)\} . \quad (5.14)
\]

Note that this map is also transitive and that \( M_{\text{noshift}}(S, T) \) is a special case of this map. Using this map, \( k_{\text{noshift}} \) is redefined to \( k'_{\text{noshift}} \):

\[
k'_{\text{noshift}}(S, T) = \sum_{(S(i), T(j)) \in M'_{\text{noshift}}(S, T)} \exp(-\beta d(S(i), T(j))^2) . \quad (5.15)
\]

Putting these ingredients together leads to the following kernel in Definition 5.2.5.

**Definition 5.2.5.** Let \( \mathcal{T} \) be a set of fixed sample rate trajectories and \( \beta > 0 \). Then we define the kernel function \( k_{\text{sumshift}} \) between two trajectories \( S, T \in \mathcal{T} \) as:

\[
k_{\text{sumshift}}^\beta(S, T) = \sum_{(S', T') \in M_{\text{sumshift}}(S, T)} k'_{\text{noshift}}(S', T') .
\]

To ensure transitivity of the map \( M_{\text{sumshift}} \) and hence PSD-ness, this kernel compares the same shifted trajectories multiple times. We can see this by the fact that comparing shift\((S, 0)\) with shift\((T, 0)\) is the same as comparing shift\((S, 1)\) and shift\((T, 1)\), etc. This means that the larger shifts, i.e. higher \( i_{\text{shift}} \), get less weight, since they are repeated less times in the mapping. We can use the fact that the same shifted trajectories are compared multiple times to speed up the computation of this kernel, since it turns computing \( |S| \text{||} |T| \) shifts into computing \( |S| + |T| \) shifts.

Another option to incorporate time shift is to compare subtrajectories of fixed length and all possible mappings between those. Note that we do consider subtrajectories here, but of a fixed length. Hence we can consider the kernel that we define as somewhere in between taking full trajectories and subtrajectories. Because we will use it in the following, we define a variant of Definition 5.2.3 that takes the product over all \( \exp(-\beta d(S(i), T(i))^2) \).

**Definition 5.2.6.** Let \( \mathcal{T} \) be a set of fixed sample rate trajectories of length \( n \) and \( \beta > 0 \). Then we define the kernel function \( k_{\text{prod}} \) between two trajectories \( S, T \in \mathcal{T} \) as:

\[
k_{\text{prod}}^\beta(S, T) = \prod_{i=1}^n \exp(-\beta d(S(i), T(i))^2) .
\]
This is a kernel by the closure under product property. The important difference with Definition 5.2.3 is that all trajectories have the same length $n$. It would also be a kernel for variable length trajectories, if we take $\exp(-\beta d(S(i), T(i))^2)$ to be 0 if one of the trajectories is not defined for $i$. However, this would result in the kernel computing 0 for two trajectories of unequal length, which is not very useful. The kernel is known as the radial basis function kernel for time-series (Rueping (2001)).

Next we define $\text{subtraj}[T, n]$ which gives all the subtrajectories of $T$ of length $n$ as:

$$\text{subtraj}[T, n] = \{ T(i, i + n - 1) \mid 0 \leq i \leq |T| - n + 1 \} .$$

With this we create the transitive map $M_{\text{subtraj}}^n(S, T)$, between all subtrajectories of length $n$ of $S$ and $T$,

$$M_{\text{subtraj}}^n(S, T) = \text{subtraj}(S, n) \times \text{subtraj}(T, n) .$$

This allows us to define the subtrajectory kernel in Definition 5.2.7.

**Definition 5.2.7.** Let $\mathcal{T}$ be a set of fixed sample rate trajectories and $\beta > 0$. Then we define the kernel function $k_{\text{fixed, sub}}$ between two trajectories $S, T \in \mathcal{T}$ as:

$$k_{\text{fixed, sub}}^{n, \beta}(S, T) = \sum_{(S', T') \in M_{\text{subtraj}}^n(S, T)} k^\beta_{\text{prod}}(S', T') .$$

If we set this $n$ to be relatively high compared to the length of a trajectory, for instance $n = \min_{T \in \mathcal{T}}(|T|)$, then only long subtrajectories are considered, therefore we say that the kernel is somewhere between a subtrajectory and full trajectory kernel. Also, the fact that $n$ is fixed limits the number of time shift that are considered. For instance, situations in which the first point of one trajectory is only compared to the last point of the other trajectory are not possible, unless $n = 1$. The computation time of this kernel is an order of magnitude worse than the previous kernel. We have to consider $|S| + |T|$ possible time shifts, which each have $|U| - n$ subtrajectories, where $|U|$ is the length of the overlap between $S$ and $T$ for a certain time shift i.e. $|U|$ is maximally $\min(|S|, |T|)$. This leads to computing similarity for $(|S| + |T|)(|U| - n)$ subtrajectories.

When we work out this definition to:

$$k_{\text{fixed, sub}}^{n, \beta}(S, T) = \sum_{(S', T') \in M_{\text{subtraj}}^n(S, T)} k^\beta_{\text{prod}}(S', T')$$

$$= \sum_{(S', T') \in M_{\text{subtraj}}^n(S, T)} \prod_{i=1}^{n} \exp(-\beta d(S'(i), T'(i))^2)$$

$$= \sum_{(S', T') \in M_{\text{subtraj}}^n(S, T)} \exp(-\beta \sum_{i=1}^{n} d(S'(i), T'(i))^2) ,$$

(5.18)
then we see a close resemblance to the soft-max kernels of Chapter 4. In this kernel we compute the soft-max of all similarities between subtrajectories of length $n$. We note that this kernel is the same as the subsequence kernel for time-series and sequences (as, e.g. by Rueping (2001)), but we derived it differently. The given definition uses $k_{prod}$ as the ‘inner’ kernel, but $k_{noshift}$ can be used instead.

**Relation to Alignments** When we consider the similarity measures $D$ as computed via a fixed sample rate, then $D_{shift}$ in Definition 5.2.4 is quite similar to the dynamic time warping (DTW) alignment. Both have computation times that are quadratic. DTW is flexible in catering for slight differences in speed by allowing repetition of elements. In $D_{shift}$ we allow for time shifts, which do not introduce an extra penalty, and hence the amount of shifting needs to be constraint. DTW also allows for some time shift, by repeating the first or last elements of a trajectory. The difference between $D_{shift}$ and edit distance is similar, but edit distance furthermore has the fixed gap penalty $g$.

The alignment methods benefit greatly from trajectory compression, because this speeds up their computation. We cannot do this in a similar fashion in the $D_{shift}$ case, since we require a high sample rate to approximate the integral. However, we can speed up computation by reducing the number of shifts to consider, by setting the shift as a multiple of the sample rate.

### 5.2.2 Subtrajectory Similarity Kernels

In the previous section we looked at kernels that considered similarity between full trajectories. Albeit that the kernel in Definition 5.2.7 already considered subtrajectories, but of a fixed length. In this section we will look at kernel versions of subtrajectory similarity.

First, we consider the version of $D$, in Definition 5.2.8, that considers subtrajectories, but no time shift.

**Definition 5.2.8.** Let $S$ and $T$ be two trajectories. And let $\tau \leq \min(t_S, t_T)$, where $t_S$ is the maximum $t$ for which $S(t)$ exists and $t_T$ is the maximum $t$ for which $T(t)$ exists. Furthermore, $\tau \geq t_{\min} > 0$. Then, we define trajectory similarity $D(S, T)$ between $S$ and $T$ as

$$D_{sub,noshift}(S, T) = \min_{\tau} \frac{\int_0^\tau d(S(t), T(t)) dt}{\tau} .$$

This variant of $D$ considers all possible subtrajectories longer than $t_{\min}$, but does not allow for time shift. This means that we require a different map than $M_{subtraj}$ for subtrajectories of length $n$:

$$M^n_{subtraj}(S, T) = \{ (S', T') \mid t_{S,0} = t_{T,0}, S'(0) = (x_{S,0}, y_{S,0}, t_{S,0}),$$

$$T'(0) = (x_{T,0}, y_{T,0}, t_{T,0}), S' \in subtraj(S, n), T' \in subtraj(T, n) \} .$$

(5.19)
In this map, which is transitive, we map the subtrajectories of \(S\) and \(T\) that have the same start time. We replace this map in \(k_{\text{fixed_sub}}\) to get:

\[
k_{\text{fixed_sub}}(S, T) = \sum_{(S', T') \in M_{\text{subtraj}}^n(S, T)} k_{\text{prod}}(S', T') .
\] (5.20)

We use this kernel to define (Definition 5.2.9) a kernel that takes the soft-max of all similarities of subtrajectories larger than \(i_{\text{min}}\).

**Definition 5.2.9.** Let \(T\) be a set of fixed sample rate trajectories and \(\beta > 0\). Then we define the kernel function \(k_{\text{sub_noshift}}\) between two trajectories \(S, T \in T\) as:

\[
k_{\text{sub_noshift}}^{i_{\text{min}}, \beta}(S, T) = \lim_{n \to \infty} \sum_{i = i_{\text{min}}}^n k_{\text{subtraj}}^{n, \beta}(S, T) .
\]

This is a positive semi-definite kernel by the closure under the limit property since the maximum length of a subtrajectory is finite. For computing this kernel we have to consider \((|T| - i_{\text{min}})^2\) subtrajectories, where \(T\) is the shortest trajectory.

The final variant of \(D\) that we look at is the one already given in Definition 5.2.1. This version considers both subtrajectories of minimum length \(t_{\text{min}}\) and time shifts. Given all that we defined above, the kernel variant of this similarity is straightforward, and we can immediately give it in Definition 5.2.10. We use the earlier defined \(k_{\text{fixed_sub}}\).

**Definition 5.2.10.** Let \(T\) be a set of fixed sample rate trajectories and \(\beta > 0\). Then we define the kernel function \(k_{\text{sub_shift}}\) between two trajectories \(S, T \in T\) as:

\[
k_{\text{sub_shift}}^{i_{\text{min}}, \beta}(S, T) = \lim_{n \to \infty} \sum_{i = i_{\text{min}}}^n k_{\text{fixed_sub}}^{n, \beta}(S, T) .
\]

Again, this kernel is PSD by the closure under the limit property. This kernel clearly has the worst running time of all the kernels, considering \((|S| + |T|)(|U| - i_{\text{min}})^2\) subtrajectories, where \(|U|\) is the length of the overlap between \(S\) and \(T\) for a certain time shift.

**Normalization and Direct Kernels of \(D\)** We normalize the positive semi-definite kernels defined above the same way as the soft-max kernels of Chapter 4, defined in Definition 5.2.11. This normalization step preserves PSD-ness.

**Definition 5.2.11.** For a set of trajectories \(T\) we compute the normalized kernel matrix \(K\) for a kernel \(k\) as:

\[
K(i, j) = \frac{k(S, T)}{\sqrt{k(S, S)k(T, T)}} ,
\]

where \(i\) and \(j\) are indexes for \(S, T \in T\).
We also want to use the different versions of \( D \) in our experiments with kernel algorithms in the next section. We turn these into kernels in Definition 5.2.12 the same way as we did in Chapter 4 with the regular versions of dynamic time warping and edit distance. Like with the kernels that we defined above, we compute \( D \) using a fixed sample rate approximation.

**Definition 5.2.12.** For a set of trajectories \( \mathcal{T} \) we compute the kernel matrix \( K_D \) for a similarity measure \( D \) as:

\[
K_D(i, j) = D(S, T),
\]

where \( i \) and \( j \) are indexes for \( S, T \in \mathcal{T} \). Furthermore, we normalize and make a kernel out of \( K_D \) by:

\[
K_D = 1 - \frac{K_D}{\min(K_D)}.
\]

We use the subscript \( D \) to distinguish the kernel matrices defined above from the kernel matrices defined for the positive semi-definite kernel functions \( k \) and matrices \( K \) defined earlier in this section.

For a summary of all the kernels that we defined above we refer to Table 10.

### 5.3 Experiments

The goal of the experiments below is to test the defined similarity measures and their corresponding kernel variants in the three typical machine learning tasks of clustering (Section 2.3.2), classification (Section 2.3.3) and outlier detection (Section 2.3.4) for vessel trajectory data. The main goal of the experiments is to determine which measures work best and if the positive semi-definite kernel variants perform better or worse.

#### 5.3.1 Experimental Set-Up

The set-up of the experiments is very similar to the experiments of Chapter 4 that investigate the different alignment kernels. Therefore, we are slightly more brief in stating the algorithm set-up for each experiment. The same datasets and algorithms are used and hence results in this section can be compared directly to the results in the previous chapter. The main difference is that the influence of trajectory compression is not investigated and that the kernels defined in the current chapter have different parameters.

We run experiments for all the kernels in Table 10. For the parameter \( n \) in \( K^{n, \beta}_{\text{sub}_1, \text{fixed}} \) and \( i_{\text{min}} \) in \( K^{i_{\text{min}}, \beta}_{\text{D,shift}}, K^{i_{\text{min}}, \beta}_{\text{D,noshift}}, K^{i_{\text{min}}, \beta}_{\text{sub,shift}}, K^{i_{\text{min}}, \beta}_{\text{sub,noshift}} \) and \( K^{i_{\text{min}}, \beta}_{\text{sub,shift}} \) we try \( \lambda = \frac{1}{3}, \frac{1}{2}, 1 \) in \( \lambda \cdot \min_{\mathcal{T} \in \mathcal{T}}(|\mathcal{T}|) \) as different settings,
i.e. for each experiment we take the length of the shortest trajectory in the dataset and set \( n \), resp. \( i_{\text{min}} \), to a fraction of that length.

To speed up the computation of the kernels that shift over the trajectory to consider different time shifts or subtrajectories, i.e. all kernels but \( K_{\text{D}_noshift} \) and \( K_{\text{noshift}} \) we introduce the parameter \( \gamma \). This \( \gamma \) determines the interval, in number of samples, over which we shift, by \( \gamma \cdot \lambda \cdot \min_{T \in \mathcal{T}}(|T|) \). For each setting of \( \lambda \) we consider two settings \( \gamma = \frac{1}{2}, 1 \). Using this \( \gamma \) keeps the positive semi-definiteness of the kernels. The smallest settings of \( \gamma = \frac{1}{2} \) and \( \lambda = \frac{1}{2} \) lead to shifts of approximately 10 samples, which is a 100-fold computation-time speed up for the slowest kernels.

For the three kernels \( K_{\text{sub}_\text{fixed}}, K_{\text{sub}_\text{noshift}} \) and \( K_{\text{sub}_\text{shift}} \) that have \( k_{\text{prod}} \) as inner kernel we also test a variant where this kernel is replaced by \( k_{\text{noshift}} \).

We use \( \beta = 1, 4, 16, 64, 256, 1024 \) for the kernels that have parameter \( \beta \). However for the kernels that make use of \( k_{\text{noshift}} \) as inner kernel, i.e. the three mentioned above and \( K_{\text{sumshift}} \) we noticed that better results are achieved by higher values for \( \beta \), here we take \( \beta = 64, 256, 1024, 4000, 64000, 256000 \).

All the kernels are computed on trajectories that have been resampled at a 0.11Hz sample rate, which is the same as we used for the no compression setting in Chapter 4.

### 5.3.2 Clustering

In the clustering task we investigate which of the kernels defined in the previous section reconstructs the gold standard of Section 2.3.2 best. We use the kernels as input in weighted kernel k-means; see Section 2.1.1.2. For each kernel we generate 10 clusterings, with \( k = 8 \), the amount in the gold standard. We evaluate the quality of a clustering with the formula from Section 2.1.2.

#### 5.3.2.1 Results

Below we present the mean score of a kernel over 10 clusterings, i.e. \( N = 10 \) for each kernel. Statistical comparisons between two kernels are done using a two-tailed Student t-test, with \( p < 0.05 \).

In the table below we present the results per kernel type. The different columns show the results of the kernels for various settings of \( \lambda \) and \( \gamma \), when applicable. The kernels \( K_{\text{D}_noshift} \) and \( K_{\text{noshift}} \) are not influenced by \( \lambda \) and \( \gamma \), thus there is only one score. Also, we note that not all columns are filled for the kernel \( K_{\text{sumshift}} \). This is because this kernel is only influenced by the \( \gamma \) parameter, which means that some combinations of \( \gamma \) and \( \lambda \) lead to the same setting, e.g. \( \gamma = \frac{1}{2}, \lambda = 1 \) and \( \gamma = 1, \lambda = \frac{1}{2} \). For the kernels that have a \( \beta \) parameter we give the results for the setting of \( \beta \) that gives the best score and we provide
the mean, minimum and maximum scores over all the β settings. We give
the mean score of 10 random clusterings, which is 0.14.

Table 11 presents the results for all the different kernels. The best
clustering score (0.80) is achieved by the $K_{D_{\text{shift}}}$ kernel under the $\lambda = 1, \gamma = 1$ setting. This score is significantly better than the best scores
for the other kernels, except for the best score for $K_{\text{sumshift}}$, which is
the best performing PSD kernel. The best clustering score (0.89) from
Chapter 4 is significantly better than the best score from Table 11.

5.3.2.2 Discussion

Like we saw with the alignment kernels in the previous chapter, the
best result is achieved by a non positive semi-definite kernel. How-
ever, the differences are not as big here as they are in that chapter.
When we consider the results for $K_{D_{\text{noshift}}}$ and $K_{D_{\text{shift}}}$ we see that al-
lowing for a little time shift ($\lambda = 1$) is beneficial for the results, but
allowing for more shift degrades performance.

For the other kernels, the different shift settings ($\lambda$ and $\gamma$) have
no effect, or only marginal (for $K_{D_{\text{subshift}}}$ and $K_{\text{subshift}}$). There is no
difference in performance for the three kernels with $k_{\text{noshift}}$ as inner
kernel, compared to their regular counterparts.

The kernels that are more expensive to compute, i.e. because they
include time shift and/or subtrajectories, show better performance
than the computationally cheap kernels $K_{D_{\text{noshift}}}$ and $K_{\text{noshift}}$.

5.3.3 Classification

In the classification task we predict the type of vessel related to a tra-
jectory in the dataset of Section 2.3.3 using a Support Vector Machine
(SVM). In this dataset we have trajectories of the four most common
vessel types. For each of these types we select 100 trajectories to cre-
ate a dataset of 400 vessel trajectories, i.e. the same dataset is used as
in the previous chapter.

As our SVM algorithm we use the C-SVC implementation of Lib-
SVM; see Section 2.1.1.1 for details. We use a 10-fold cross validation
set-up to evaluate the different kernels, where we compute the clas-
sification accuracy for each fold. Within each fold the C parameter is
optimized using, again, 10-fold cross validation.

5.3.3.1 Results

We present the mean classification accuracy over 10 folds, i.e. $N = 10$
for each kernel. Statistical comparisons between the results of two ker-
nels are done using a two-tailed paired t-test with $p < 0.05$, since the
10 folds are the same for each kernel. The table presenting the results
below has the same structure as used for the clustering experiment.
The mean random classification accuracy is 25%.
Table 11: Mean value of clus_score for different integral over time kernels.

<table>
<thead>
<tr>
<th></th>
<th>$\lambda = \frac{1}{4}$</th>
<th>$\gamma = \frac{1}{4}$</th>
<th>$\lambda = \frac{1}{2}$</th>
<th>$\gamma = \frac{1}{2}$</th>
<th>$\lambda = 1$</th>
<th>$\gamma = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_{p_{noshift}}$</td>
<td>0.62</td>
<td>0.59</td>
<td>0.65</td>
<td>0.75</td>
<td>0.74</td>
<td>0.80</td>
</tr>
<tr>
<td>$K_{p_{shift}}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$K_{sub_{noshift}}$</td>
<td></td>
<td>0.53</td>
<td>0.54</td>
<td>0.57</td>
<td>0.57</td>
<td>0.59</td>
</tr>
<tr>
<td>$K_{sub_{shift}}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$K_{p_{shift}}$</td>
<td>0.58</td>
<td>0.69</td>
<td>0.66</td>
<td>0.69</td>
<td>0.66</td>
<td>0.69</td>
</tr>
</tbody>
</table>

- mean
  - max
  - min

- fixed
  - sub
  - shift

- noshift
  - best ($\beta = 64$)
  - with $\beta = 256$
  - with $\beta = 16$
  - with $\beta = 1$
  - with $\beta = 64$

- sub
  - shift

- noshift
  - with $\beta = 64$

53 EXPERIMENTS
We present the classification results for all the different kernels in Table 12. The best accuracy (68.25) is achieved by the $K_{\text{sumshift}}$ kernel. This accuracy is significantly better than the accuracies for the different $K_D$ kernels, but not significantly better than the accuracies for the other PSD kernels. The best accuracy (76.25) achieved for the alignment kernels is significantly better than the best result achieved in this chapter.

5.3.3.2 Discussion

Contrary to the alignment kernels of Chapter 4, the best results are not achieved by the non-PSD kernels, but by the positive semi-definite kernels, by a large margin. For almost all the kernels the influence of the $\lambda$ and $\gamma$ settings is nearly non-existent. The exception being the $K_{D_{\text{sub_shift}}}$ kernel, which performs somewhat better with higher $\lambda$. Like we saw in the clustering experiment, there is not much difference between the performance of the three kernels with $k_{\text{noshift}}$ as their inner kernel, and the performance of their corresponding regular versions. Similar to the results for alignment kernels, it does not pay off to compute more expensive kernels, since the $K_{D_{\text{noshift}}}$ kernel does not perform worse than the more expensive $K_D$ kernels, and the same holds for the $K_{\text{noshift}}$ kernel.

5.3.4 Outlier Detection

In the outlier detection task we use the kernels with one-class Support Vector Machines to find the outliers among a set of trajectories of cargo ships and tankers; see Section 2.3.4. This set consists of 786 trajectories, 39 of which are outliers.

We use the one-class version of SVM in LibSVM. We randomly split the 747 normal trajectories into a training set of $\frac{3}{4}$ and a test set of the rest. To this test set we add the outlying trajectories. We do this split 10 times per kernel. For each split we train a one-class SVM model. The $\nu$ parameter is optimized during training using 10-fold cross validation. The decision values that this model gives on the test set are used to generate a ranking of the test trajectories. On this ranking, we use the precision@39 measure to evaluate the performance of the different kernels.

5.3.4.1 Results

The mean precision@39 scores for each kernel are computed over 10 splits, thus $N = 10$. Statistical comparisons between two kernels are done using a two-tailed Student t-test, with $p < 0.05$. The following table presents the result in the same way as for the clustering and classification experiments. The mean precision@39 for 10 random rankings of the test set is 0.16.
Table 12: Mean accuracy (%) for different integral over time kernels.

<table>
<thead>
<tr>
<th>Kernel Type</th>
<th>Configuration</th>
<th>λ = (\frac{1}{4})</th>
<th>λ = (\frac{1}{2})</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ = (\frac{1}{4})</td>
<td>γ = (\frac{1}{2})</td>
<td></td>
</tr>
<tr>
<td>K_{D_{noshift}}</td>
<td>49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K_{D_{shift}}</td>
<td>44.25</td>
<td>44.25</td>
<td></td>
</tr>
<tr>
<td>K_{D_{sub_noshift}}</td>
<td>39.25</td>
<td>40.25</td>
<td></td>
</tr>
<tr>
<td>K_{D_{sub_shift}}</td>
<td>43.75</td>
<td>43.75</td>
<td></td>
</tr>
<tr>
<td>K_{p2shift}</td>
<td>best (β = 1024)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>65.5</td>
<td>65.5</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>58.21</td>
<td>58.25</td>
<td></td>
</tr>
<tr>
<td>K_{noshift}</td>
<td>best (β = 4000)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.25</td>
<td>67.25</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.58_{67.75}^{60.25}</td>
<td>64.38_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{fixed_sub}</td>
<td>best (β = 256)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.83_{67.75}^{60.25}</td>
<td>64.36_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{sub_noshift}</td>
<td>best (β = 64)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.83_{67.75}^{60.25}</td>
<td>64.36_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{sub_shift}</td>
<td>best (β = 256)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.25</td>
<td>67.25</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.96_{67.75}^{60.25}</td>
<td>64.63_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{sub_shift}</td>
<td>with (β = 4000)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.88_{67.75}^{60.25}</td>
<td>64.88_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{fixed_sub}</td>
<td>with (β = 4000)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.88_{67.75}^{60.25}</td>
<td>64.88_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{sub_noshift}</td>
<td>with (β = 64000)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.96_{67.75}^{60.25}</td>
<td>64.96_{67.75}^{60.25}</td>
<td></td>
</tr>
<tr>
<td>K_{sub_shift}</td>
<td>with (β = 64000)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mean_{max}</td>
<td>67.75</td>
<td>67.75</td>
<td></td>
</tr>
<tr>
<td>mean_{min}</td>
<td>64.88_{67.75}^{60.25}</td>
<td>64.88_{67.75}^{60.25}</td>
<td></td>
</tr>
</tbody>
</table>
Table 13 presents the results for the outlier detection experiment. The best score (0.53) is achieved by the $K_{sub\_noshift}$ kernel. This score is significantly better than all the other best scores, except for the best score (0.49) for the $K_{noshift}$ kernel and the best score (0.51) for the $K_{sub\_noshift}$ with $k_{noshift}$ kernel. However, these scores are significantly worse than the best score (0.84) for the alignment kernels.

5.3.4.2 Discussion

Like we saw with the soft-max kernels in Chapter 4, the PSD kernels perform a lot better on the outlier detection task than the non-PSD kernels. We see, as in the previous two experiments, that the influence of the shifting parameters $\lambda$ and $\gamma$ is marginal. Also similar to the previous two experiments, the three kernels with $k_{noshift}$ as inner kernels do not differ much in performance from their regular counterparts. The not significantly worse performance of the cheap $K_{noshift}$ kernel does not warrant the computation of more expensive kernels on the outlier detection task.

The rather large performance difference between the kernels defined in this chapter and the alignment kernels from Chapter 4 for the outlier detection task is striking. This might be due to the fact that piecewise linear segmentation ‘magnifies’ the strange part in a trajectory, since this strange part adds extra retained points, compared to a regular trajectory. This effect leads to bad alignments. However, in the fixed sample rate, integral based approach, the strange part is not particularly ‘magnified’.

5.4 Conclusions & Future Work

The main result for this chapter is that kernels based on integrating the distance between two trajectories over time perform worse than the kernels based on alignment measures from Chapter 4. In all the three experiments we obtained significantly worse results than achieved in the previous chapter. Moreover, apart from the most basic kernels ($K_{D\_noshift}$ and $K_{noshift}$) these integral based kernels are more expensive to compute, since they do not benefit from trajectory compression, i.e. they require a large number of samples to approximate the integral.

Like in the previous chapter we see that for the different tasks, different kernels work best. Especially noteworthy are the results of the outlier detection experiment, where the concept of soft-max works reasonably well, as it also did for the soft-max alignment kernels.

For future work it would be interesting to investigate in which kinds of datasets and tasks the performance of the alignment measures breaks down and where the more precise approach of computing the integral gives better results. In the same vein it would be interesting
Table 13: Precision@39 for different integral over time kernels.

<table>
<thead>
<tr>
<th></th>
<th>$\lambda = \frac{1}{4}$</th>
<th>$\lambda = \frac{1}{2}$</th>
<th>$\lambda = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma = \frac{1}{4}$</td>
<td>0.12</td>
<td>0.12</td>
<td>0.14</td>
</tr>
<tr>
<td>$\gamma = \frac{1}{2}$</td>
<td>0.13</td>
<td>0.13</td>
<td>0.17</td>
</tr>
<tr>
<td>$\gamma = 1$</td>
<td>0.16</td>
<td>0.15</td>
<td>0.15</td>
</tr>
</tbody>
</table>

| $k_d_{noshift}$    | 0.23                     | 0.23                      | 0.23           |
| $k_d_{shift}$      | 0.12                     | 0.13                      | 0.14           |
| $k_{dub\_noshift}$| 0.17                     | 0.16                      | 0.18           |
| $k_{dub\_shift}$  | 0.16                     | 0.15                      | 0.15           |

| $k_{noshift}$      | best ($\beta = 256$)    | 0.49                      | 0.49           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.36 $\frac{\gamma}{\gamma}$ | 0.36 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.25 $\frac{\gamma}{\gamma}$ | 0.25 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 1024$)    | 0.46                      | 0.46           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.40 $\frac{\gamma}{\gamma}$ | 0.40 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.26 $\frac{\gamma}{\gamma}$ | 0.26 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 64$)      | 0.38                      | 0.38           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 16$)      | 0.49                      | 0.49           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.41 $\frac{\gamma}{\gamma}$ | 0.41 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.28 $\frac{\gamma}{\gamma}$ | 0.28 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 64$)      | 0.38                      | 0.38           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 4000$)    | 0.37                      | 0.37           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ | 0.17 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 1024$)    | 0.49                      | 0.49           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.40 $\frac{\gamma}{\gamma}$ | 0.40 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.18 $\frac{\gamma}{\gamma}$ | 0.18 $\frac{\gamma}{\gamma}$ |
| $k_{noshift}$      | best ($\beta = 4000$)    | 0.38                      | 0.38           |
|                   | max $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ | 0.30 $\frac{\gamma}{\gamma}$ |
|                   | min $\frac{\gamma}{\gamma}$ | 0.18 $\frac{\gamma}{\gamma}$ | 0.18 $\frac{\gamma}{\gamma}$ |
to research what tasks and datasets are advantageous for the positive semi-definite kernels, compared to the non-PSD kernels.