Making Kernel Density Estimation Robust towards Missing Values in Highly Incomplete Multivariate Data without Imputation

Richard Leibrandt* Stephan Günnemann†

Abstract

Density estimation is one of the most frequently used data analytics techniques. A major challenge of real-world datasets is missing values, originating e.g. from sampling errors or data loss. The recovery of these is often impossible or too expensive. Missing values are not necessarily limited to a few features or samples, rendering methods based on complete auxiliary variables unsuitable. In this paper we introduce three models able to deal with such datasets. They are based on the new concept of virtual objects. Additionally, we present a computationally efficient approximation. Generalizing KDE, our methods are called Warp-KDE. Experiments with incomplete datasets show that Warp-KDE methods are superior to established imputation methods.

1 Introduction

Density estimation methods aim to determine the density in a data space, which is often interpreted as an estimate of the probability density function (PDF) from which the data elements are sampled from. While the statistics community tend to focus on density estimation for univariate data, in the data mining community techniques are expected to handle multivariate data objects. Thus, in this work we revise multivariate density estimation for elements represented as multi-dimensional vectors that indicate the attributes (features) of the data objects. Particularly, we extend the approach of kernel density estimation (KDE).

Regarding data mining algorithms there are three general directions of research that are of much interest: effectiveness, efficiency, and versatility. While effectiveness and efficiency of KDE are well-studied topics in literature, in this work we aim to improve versatility regarding incompleteness.

1.1 Incomplete data. An important issue of modern data mining is how to deal with incomplete data. Reasons can be failing sensors (e.g. in factories), loss of data (e.g. in big data systems), sampling of too few features (e.g. in medical datasets), or merging of datasets with different features (e.g. in astronomy). Often it is either financially too expensive or impossible to retrieve those missing values. Despite the large number of publications on density estimation, literature addressing the missing value problem is comparatively sparse.

Generally there are four basic approaches to deal with missing values: *Feature marginalization* ignores incomplete features. *Object marginalization*, also known as complete case analysis [4] or whole data strategy [11], ignores incomplete objects. Both result in high information loss. *Imputation* estimates missing values from available values, e.g. [20], before the desired data mining method is performed. Since imputed data is less reliable than actually observed data [22], it may lead to wrong conclusions, especially if a feature has many missing values. *Integrative methods* are designed to deal with missing values directly without preprocessing, e.g. [9]. They are the best option if applicable. The contribution of this paper is such an integrative method.

In literature three types of missing data are particularly often mentioned that characterize what influences the locations of missing values [5, 12, 15]: (i) Data is not missing at random (NMAR) if the places depend on the missing values themselves (and possibly observed data), e.g. high-income earners not disclosing their income (*income* being the feature). (ii) Data is missing at random (MAR) if the places only depend on observed values, e.g. young people with no income leave *income* open (observed *age* being responsible). (iii) Data is missing completely at random (MCAR) if the places do not depend on any values, e.g. some interviewers forgot to ask for *income*, then questionnaires are merged.

On the one hand, we often do not have access to complete variables. In these cases, MAR approaches are not feasible. On the other hand, in many use cases it is more reasonable to assume MCAR than MAR, especially in industrial settings. For example, when datasets with naturally different features are merged or datasets that should have the same features, but some sources failed to collect certain features. Consider the mentioned interviewer, or medical examiners that do not record everything, or sensors that randomly fail in manufacturing facilities.

*Technical University of Munich, r.leibrandt@tum.de
†Technical University of Munich, guennemann@in.tum.de

1.2 Parametric Density Estimation of MCAR Data. Since our method is integrative, we give an overview of integrative methods for density estimation of incomplete data. Two major approaches have been adapted: parametric and non-parametric.

Parametric density estimation assumes the data to originate from a process that can be described with known functions which are parametrized to fit the data, often using the expectation-maximization (EM) principle. The most common example being a mixture of Gaussians and extensions thereof [8]. The standard approach to fit the models to incomplete data integratively is *iterative imputation*. Its basic idea is to have a third step, besides the E- and M-steps, in which the missing values are imputed, usually from other data objects that are considered to originate from the same model at that iteration. Parametric approaches have the issues to require a fixed model class before the fitting, which can greatly effect the outcome, and, thus, not being flexible in building arbitrary density distributions. Furthermore, iterative imputation influences the imputation in the next step [18], distorting the results.

1.3 Non-parametric KDE and Kernel Extensions for MAR Data. Non-parametric density estimation models the data objects with functions, called kernels, which are summed up to the overall PDF. Intuitively, this approach considers each data object as a source of a small random process, using as many models as there are data objects. Instead of optimizing parameters to fit models to the data, non-parametric approaches are directly driven by the geometry of the data objects.

A popular non-parametric approach is kernel density estimation (KDE). KDE offers numerous advantages: (i) It adapts to different kinds of data topologies. (ii) It does not require a number of models a priori, which is difficult to find, especially in data exploration. (iii) It is deterministic without depending on an initialization.

The problem of missing values for kernel-based methods has been focused on MAR data: A kernel extension for incomplete univariate data is presented in [17]. [14] expands on [17] for the multivariate case, but only for binary values. In contrast, our methods can deal with multivariate, real-valued data.

For KDE the most widely used approach for incomplete data is for the univariate case, assuming that the position where values are missing depends on auxiliary variables, which must be complete [4]. Differently, [19] requires some data objects to be complete. It focuses on bivariate data, although an extension for multivariate data is presented. In contrast, our methods can deal with multivariate data in which all features and all data objects have missing values.

1.4 Contribution. This paper contributes integrative non-parametric methods for estimating densities of multivariate highly incomplete datasets with values missing completely at random (MCAR). To our knowledge, no comparative non-parametric methods exist.

To claim a method is able to deal with highly incomplete data, we suggest it should have no restrictions regarding number or position of missing values. In other words, it should be able to accept datasets which have missing values in each feature and in each data object used. Available integrative non-parametric density estimations discuss MAR datasets, which rely on complete *auxiliary variables*. We show that we are able to handle data with up to 50% missing values, which is highly incomplete compared to real-life data.

Furthermore, our approach is fully integrative, making no assumptions about missing values with imputation or losing information due to marginalization. Our approach is not restricted to binary or univariate data, but also handles multivariate, real-valued data.

We achieve this goal by extending KDE, providing a new location-sensitive kernel for incomplete data objects, called the Warp-kernel. Each object is associated with an individual Warp-kernel. This concept is in strong contrast to classical KDE, where each object uses the same kernel. We show that our integrative methods are superior to pre-processing with established imputation.

2 Warp-kernels based on Virtual Objects

In classical KDE, each data object is modeled with a Gaussian kernel. This is only possible if the data objects have no missing values (determined objects). In the following we are going to present kernels, so called Warp-kernels, that model objects which do have missing values (undetermined objects).

First, in Sec. 2, we present three Warp-kernels which are based on estimating the density for undetermined objects with the help of so called virtual objects. Then, in Sec. 3, we present two approaches to approximate the previous concepts. Finally, in Sec. 4, we compare the three accurate methods with their approximations and pre-processing the data with imputation.

2.1 Notations. This paper focuses on data objects $\boldsymbol{o}_i \in \mathcal{O} \subset (\mathbb{R} \cup \{\varnothing\})^F$ whose features are either real numbers ($\mathbb{R}$) or missing values ($\varnothing$). We define $\varnothing \cdot 0 = 0$. Finite sets are printed in calligraph (e.g. $\mathcal{O}$), infinite sets in blackboard bold (e.g. $\mathbb{R}$). The features span the real coordinate space $\mathbb{R}^F$ (feature space) with F being the number of features and with $\boldsymbol{x} \in \mathbb{R}^F$ being any point in the feature space. We denote the f-th coordinate of a data object $\boldsymbol{o}_i$ as $o_{i:f}$. The scalar $x_{:f}$ is the f-th coordinate of $\boldsymbol{x}$ with $f \in \mathcal{F}$, $\mathcal{F} = \{1, ..., F\} \subset \mathbb{N}$.

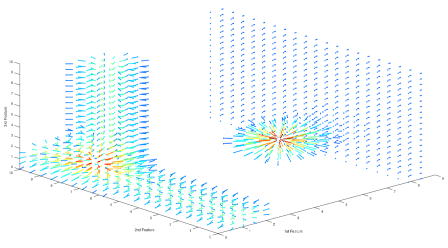


Fig. 1. Gradients of $\mathrm{WDF}_{\text{tube}}$ for objects $(1, ⦰, 1)$, $(2, 8, ⦰)$, $(5, 3, 4)$, $(8, ⦰, ⦰)$. Illustrates $\mathbb{V}_u$ for different missing values.

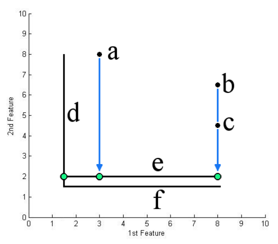


Fig. 2. black: $\mathbb{V}_i \cap \mathbb{D}$, blue: projection, green: $\mathcal{O}_{\text{``e''}}^{\leftarrow}$

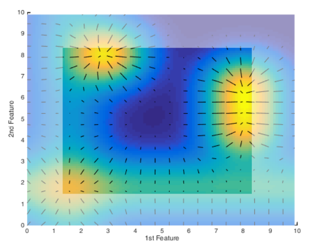


Fig. 3. $\mathrm{WDF}_{\text{tube}}$ of Fig. 2 data. Intense colors mark $\mathbb{D}$.

An object is denoted as a determined object $\boldsymbol{o}_d \in \mathbb{R}^F$ if all features of the object are available. All determined objects form the set $\mathcal{O}_{\text{d}}$. An object is denoted as an undetermined object $\boldsymbol{o}_u \in (\mathbb{R} \cup \{⦰\})^F$ if some features are missing. Undetermined objects form $\mathcal{O}_{\text{u}}$. For the i-th object $\boldsymbol{o}_i$, available features form the set $\mathcal{A}_i = \{\, f \in \mathcal{F} \mid o_{i:f} \neq ⦰ \,\}$ and missing features the set $\mathcal{M}_i = \{\, f \in \mathcal{F} \mid o_{i:f} = ⦰ \,\}$, thus $\mathcal{A}_i \cup \mathcal{M}_i = \mathcal{F}$. Objects coming from the actual data are denoted as real objects $\boldsymbol{o}_r$, forming $\mathcal{O}_{\text{r}}$. Later we will introduce virtual objects $\boldsymbol{o}_v$, forming $\mathcal{O}_{\text{v}}$. The cardinality of a set is denoted with $|\cdot|$, e.g. $|\mathcal{O}|$. The empty set is denoted with $\varnothing$.

2.2 Information Mass and Warp-kernel for Complete Datasets. The non-parametric approach KDE models data objects with functions, so called kernels. For multivariate data, a common kernel is the multivariate, symmetric Gaussian kernel

$$\text{(2.1)} \qquad \mathrm{G}\left(\boldsymbol{x} \mid \boldsymbol{o}_i, \omega\right) = \frac{1}{\sqrt{(2 \cdot \pi \cdot \omega)^F}} \cdot \exp\left(\frac{\|\boldsymbol{x} - \boldsymbol{o}_i\|_2^2}{-2 \cdot \omega}\right)$$

with $\boldsymbol{o}_i$ and ω being mean and bandwidth of the Gaussian and F the dimensionality of the feature space. KDE considers the density of the data equal to the sum of the kernels with $\mathrm{PDF}(\boldsymbol{x}) = |\mathcal{O}|^{-1} \cdot \sum_{i=1}^{|\mathcal{O}|} \mathrm{G}\left(\boldsymbol{x} \mid \boldsymbol{o}_i, \omega\right)$. The estimated PDF enjoys multiple desirable properties, e.g. differentiability, which PDFs based on other kernels do not generally share [2]. Thus, we chose the Gaussian kernel, to be extended to the Warp-kernels. Before laying out how, we motivate the notion of information mass.

Each of the objects $\boldsymbol{o}_i$ is considered to have an information mass $m_i \in \mathbb{R}_{>0}$. From a probabilistic point of view, m_i adds probability (or evidence) of presence to the feature space in the region near the object's location $\boldsymbol{o}_i$. A Warp-kernel describes what "near" means and how m_i is distributed in the feature space $\mathbb{R}^F$. For complete datasets, we can formulate the Warp-kernel and Warp Density Function (WDF) with

$$\text{(2.2)} \qquad \mathrm{W}_{\text{cpl},i}\left(\boldsymbol{x}\right) \coloneqq \frac{m_i}{\sqrt{(2 \cdot \pi \cdot \omega)^F}} \cdot \exp\left(\frac{\|\boldsymbol{x} - \boldsymbol{o}_i\|_2^2}{-2 \cdot \omega}\right) ,$$

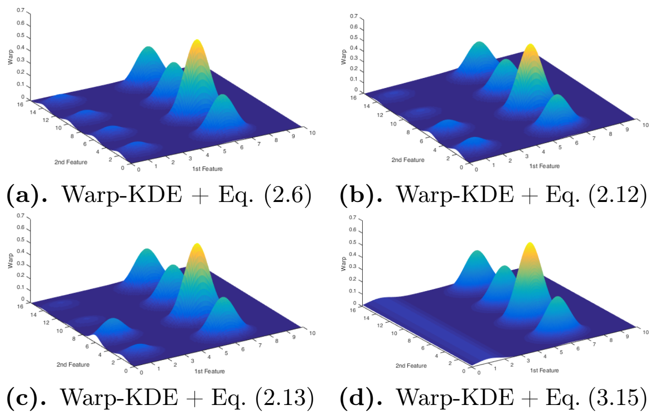
$$\text{(2.3)} \qquad \mathrm{WDF}_{\text{cpl}}\left(\boldsymbol{x}\right) \coloneqq \sum\nolimits_{i=1}^{|\mathcal{O}|} \mathrm{W}_{\text{cpl},i}\left(\boldsymbol{x}\right) .$$

The difference to the Gaussian PDF is the introduced mass and that the integral of WDF over $\mathbb{R}^F$ ($\mathbb{D} \subset \mathbb{R}^F$ from Sec. 3.2) is the sum of all mass instead of 1. By allocating different m_i to different objects, objects can be differentiated in importance. E.g., differently reliable sensors would add different amounts of mass to $\mathbb{R}^F$. In Sec. 2.4 we are going to distribute the mass among substitutes of undetermined objects.

2.3 Kernels built from "inner KDEs". Just as we can interpret the kernel of a determined object as its probability of presence, modeling the uncertainty of its measurement, we want to model undetermined objects $\boldsymbol{o}_u$ so that we can apply the same interpretation. The notion of the following Warp-kernels views each undetermined object $\boldsymbol{o}_u$ as being a determined object originally; its missing values exist in principle, but got lost. We can be certain that the $\boldsymbol{o}_u$ is an element of the linear manifold

$$\text{(2.4)} \qquad \mathbb{V}_i = \{\, \boldsymbol{x} \in \mathbb{R}^F \mid x_{:f} = o_{i:f} \quad \forall f \in \mathcal{A}_i \,\} .$$

For a determined object $\boldsymbol{o}_d$ this means, $\mathbb{V}_d$ is simply $\boldsymbol{o}_d$ itself – describing a zero-dimensional point in the feature space $\mathbb{R}^F$. If an undetermined object $\boldsymbol{o}_u$ is missing one value, $\mathbb{V}_u$ is a straight line, if it misses two values, $\mathbb{V}_u$ is a plane, and so on, illustrated in Fig. 1. Fig. 2 displays three determined objects and three with one missing value in black. For a determined object, the density distribution is the Gaussian-shaped Warp-kernel $\mathrm{W}_{\text{cpl},i}$, but what is the density distribution of an undetermined object? We want to answer the question of how an undetermined object contributes to the WDF by considering the linear manifold $\mathbb{V}_u$ as the basis, namely as the space for yet another KDE – a nested, inner KDE so to speak – which, as a whole, is the Warp-kernel for the undetermined object. We can interpret the Warp-kernel for $\boldsymbol{o}_u$ as the probability of presence of $\boldsymbol{o}_u$. All vectors, from which this Warp-kernel is built, need to be elements of the linear manifold $\mathbb{V}_u$, but the Warp-kernel must be defined in the entire feature space $\mathbb{R}^F$, leaving us with the question from which vectors to build it.



(a). Warp-KDE + Eq. (2.6) **(b).** Warp-KDE + Eq. (2.12)

(c). Warp-KDE + Eq. (2.13) **(d).** Warp-KDE + Eq. (3.15)

Fig. 4. Kernels for the objects $(6, 2)$, $(6.1, 10)$, $(6.1, 14)$, $(1, \obslash)$ with mass $m_i = 1$ and $(6, 6)$ with $m_i = 2$.

Algorithm 1 Derivation of virtual determined objects

```
 1: split $\mathcal{O}$ into $\mathcal{O}_\mathrm{d}$ and $\mathcal{O}_\mathrm{u}$
 2: $\mathcal{O}_{\mathrm{u,NEW}} \coloneqq \mathcal{O}_\mathrm{u}$
 3: while $\mathcal{O}_{\mathrm{u,NEW}} \neq \varnothing$ do
 4:     $\mathcal{O}_{\mathrm{u,ACT}} \coloneqq \varnothing$
 5:     for all $\boldsymbol{o}_u \in \mathcal{O}_{\mathrm{u,NEW}}$ do
 6:         for all $\boldsymbol{o}_p \in \mathcal{O}_\mathrm{d} \cup \mathcal{O}_{\mathrm{u,NEW}}$ do
 7:             derive location and mass Eq. (2.5)
                and (2.6) for new $\boldsymbol{o}_{v,\mathrm{NEW}} \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u$
 8:             if $\boldsymbol{o}_{v,\mathrm{NEW}}$ is determined then
 9:                 add $\boldsymbol{o}_{v,\mathrm{NEW}}$ to $\mathcal{O}_\mathrm{d}$
10:             else
11:                 add $\boldsymbol{o}_{v,\mathrm{NEW}}$ to new $\mathcal{O}_{\mathrm{u,ACT}}$
12:     $\mathcal{O}_{\mathrm{u,NEW}} \coloneqq \mathcal{O}_{\mathrm{u,ACT}}$
13: return $\mathcal{O}_\mathrm{d}$
```

For $F = 2$, the Warp-kernel can be built from the projections of valid projector objects $\boldsymbol{o}_p$ on the $\mathbb{V}_u$. An object $\boldsymbol{o}_p$ is a valid projector object if it has at least all those features available that are missing for $\boldsymbol{o}_u$. E.g., for the object e in Fig. 2 the Warp-kernel is built from the projections of a, b, c, d (but not f) on e. From the resulting vectors (green in Fig. 2) this "inner KDE", the Warp-kernel, can be built. In Fig. 4a, another example is displayed, where the small hills together are such a Warp-kernel. Note how the locations of the small hills are specified by the second features of the large hills.

For $F > 2$ we do not necessarily have such valid projector objects available. E.g., we cannot build the Warp-kernel for the object $(0, \obslash, \obslash)$ from the other objects $(\obslash, 1, \obslash)$, $(\obslash, \obslash, 2)$, $(\obslash, \obslash, 3)$ in such a way, since none of them qualifies as "valid projector objects". However, with the introduction of virtual objects, projectors are already valid if they have at least one feature available that is missing for $\boldsymbol{o}_u$.

2.4 Virtual Objects.

The basic idea of virtual objects is to split a real undetermined object $\boldsymbol{o}_u$ with high dimensional $\mathbb{V}_u$ (e.g. "e" in Fig. 2) into multiple virtual objects $\boldsymbol{o}_v$ (displayed in green), that form the set $\mathcal{O}_u^{\leftarrow}$, with $\mathbb{V}_u$ of lower dimensionality. E.g. in a $\mathbb{R}^3$, by projecting the objects $\boldsymbol{o}_{p_1} = (\obslash, 1, \obslash)$, $\boldsymbol{o}_{p_2} = (\obslash, \obslash, 2)$, $\boldsymbol{o}_{p_3} = (\obslash, \obslash, 3)$ on $\boldsymbol{o}_u = (0, \obslash, \obslash)$, $\boldsymbol{o}_u$ is split into the virtual objects $\mathcal{O}_u^{\leftarrow} = \{(0, 1, \obslash), (0, \obslash, 2), (0, \obslash, 3)\}$. Note that the new objects in $\mathcal{O}_u^{\leftarrow}$ have now $\mathbb{V}_i$ of lower dimensionality (lines) than $\mathbb{V}_u$ of $\boldsymbol{o}_u$ (which is a plane). So, a virtual object $\boldsymbol{o}_v$ is the projection of an object $\boldsymbol{o}_p$ onto an undetermined object's $\mathbb{V}_u$. An allegory could be the projection of an image ($\boldsymbol{o}_p$) onto a canvas ($\boldsymbol{o}_u$), hence the following notation as canvas Warp-kernel. We say "$\boldsymbol{o}_v$ is directly derived from the object $\boldsymbol{o}_u$ via the projector object $\boldsymbol{o}_p$", which we denote with $\boldsymbol{o}_v \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u$. The virtual object's feature values are those of $\boldsymbol{o}_u$ if they are available from $\boldsymbol{o}_u$, otherwise of $\boldsymbol{o}_p$. If a feature is missing from both, the feature is also missing from $\boldsymbol{o}_v$.

If a virtual object is still undetermined, further recursive splitting is applied until all virtual objects are determined. E.g. $(0, 1, \obslash)$ is further split into the objects of location $(0, 1, 2)$ and $(0, 1, 3)$. Thus, a virtual determined object $\boldsymbol{o}_d$ might originate from a real undetermined object $\boldsymbol{o}_u$ via a chain of direct derivations. The set of all $\boldsymbol{o}_d$ that originate from $\boldsymbol{o}_u$ is denoted with $\mathcal{O}_u^{\leftarrow}$. If the real object $\boldsymbol{o}_r$ is determined, $\mathcal{O}_r^{\leftarrow} = \{\boldsymbol{o}_r\}$. For now, the mass of $\boldsymbol{o}_u$ is equally distributed among the virtual objects directly derived from it. Formally

$$(2.5)\quad o_{v:f} \coloneqq \begin{cases} o_{u:f} & \text{if } f \in \mathcal{A}_{\mathrm{u},i} \\ o_{p:f} & \text{if } f \in \mathcal{M}_{\mathrm{u},i} \end{cases}, \qquad \forall\ \boldsymbol{o}_v \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u ,$$

$$(2.6)\quad m_v \coloneqq \frac{m_u}{|\mathcal{O}_u^{\leftarrow}|}, \qquad \forall\ \boldsymbol{o}_v \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u .$$

Algorithm 1 implements the recursive derivation of determined objects. Using the virtual objects $\boldsymbol{o}_d \in \mathcal{O}_i^{\leftarrow}$, we formulate the Warp-kernel for $\boldsymbol{o}_i$ and the WDF as

$$(2.7)\quad \mathrm{W}_{\mathrm{canv},i}(\boldsymbol{x}) \coloneqq \sum_{\boldsymbol{o}_d \in \mathcal{O}_i^{\leftarrow}} \frac{m_d}{\sqrt{(2 \cdot \pi \cdot \omega)^F}} \cdot \exp\left(\frac{\|\boldsymbol{x} - \boldsymbol{o}_d\|_2^2}{-2 \cdot \omega}\right)$$

$$(2.8)\quad \mathrm{WDF}_{\mathrm{canv}}(\boldsymbol{x}) \coloneqq \sum\nolimits_{i=1}^{|\mathcal{O}_\mathrm{r}|} \mathrm{W}_{\mathrm{canv},i}(\boldsymbol{x}) .$$

Instead of defining the Warp-kernels for the real undetermined objects and then adding them up with the Warp-kernels for the real determined object, we can define the WDF a lot more compactly by adding up all determined objects, virtual and real, with the formula

$$(2.9)\quad \mathrm{WDF}_{\mathrm{canv}}(\boldsymbol{x}) \coloneqq \sum_{\boldsymbol{o}_d \in \mathcal{O}_\mathrm{d}} \frac{m_j}{\sqrt{(2 \cdot \pi \cdot \omega)^F}} \cdot \exp\left(\frac{\|\boldsymbol{x} - \boldsymbol{o}_d\|_2^2}{-2 \cdot \omega}\right).$$

2.5 Making Warp-kernel Projector-Sensitive.

The canvas Warp-KDE in Sec. 2.3 considers the locations of the virtual objects, as demonstrated in Fig. 4a, but it does not consider how far a projector is from $\mathbb{V}_u$: Four virtual kernels emerge, each of the same size, since the virtual objects' masses are equal, although the projectors have different distances from $\mathbb{V}_u$. Considering that the Warp-kernel $\mathrm{W}_{\mathrm{canv},i}(u)$ should describe the probability of presence of $\boldsymbol{o}_u$, we argue that a projector $\boldsymbol{o}_{p_1}$ closer to $\mathbb{V}_u$ should lead to a larger $\mathrm{W}_{\mathrm{canv},u}(\boldsymbol{o}_{v_1})$ (where $\boldsymbol{o}_{v_1} \xleftarrow{\boldsymbol{o}_{p_1}} \boldsymbol{o}_u$) than a distant projector $\boldsymbol{o}_{p_2}$ at $\mathrm{W}_{\mathrm{canv},u}(\boldsymbol{o}_{v_2})$. This motivates us to take the distance between an undetermined object $\boldsymbol{o}_u$ and its projectors into account. The distance between $\boldsymbol{o}_i$ and $\boldsymbol{o}_p$ is defined by

$$\text{(2.10)} \quad \operatorname{dist}(\boldsymbol{o}_i, \boldsymbol{o}_p) \coloneqq \left\|\boldsymbol{e}_i \circ \boldsymbol{e}_p \circ \boldsymbol{o}_i - \boldsymbol{e}_i \circ \boldsymbol{e}_p \circ \boldsymbol{o}_p\right\|_2 \ ,$$

$$\text{(2.11)} \quad e_{j:f} \coloneqq \begin{cases} 1\ , & \text{if } o_{j:f} \neq \varnothing \\ 0\ , & \text{if } o_{j:f} = \varnothing \end{cases} \ ,$$

where $\circ$ is the element-wise Hadamard product.

We alter the mass derivation Eq. (2.6) to

$$\text{(2.12)} \quad m_v \coloneqq m_u \cdot \frac{\exp\left(\frac{\operatorname{dist}(\boldsymbol{o}_u,\boldsymbol{o}_p)^2}{-2\cdot\omega}\right)}{\sum_{\boldsymbol{o}_q \in \mathcal{O}_u^{\leftarrow}} \exp\left(\frac{\operatorname{dist}(\boldsymbol{o}_u,\boldsymbol{o}_q)^2}{-2\cdot\omega}\right)} \ ,$$

for $\boldsymbol{o}_v \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u$ by distributing the mass of the object $\boldsymbol{o}_u$ depending on the distance to the projector object. We use a similarity measure based on the Gaussian to take the Gaussian influence of the projector object $\boldsymbol{o}_p$ on the manifold $\mathbb{V}_u$ into account. In Fig. 4b we see the effect of taking the distance to the projector object into account, where the two upper determined objects are further away from $\boldsymbol{o}_u$ than the two lower.

We expand further: In Fig. 4b we see that even though the projectors have different masses, this does not effect the hills of the virtual objects. We argue that this should be the case, since an object's mass describes how likely it is that new objects would appear in its region. This motivates us to take the mass of the projectors into account when distributing $\boldsymbol{o}_u$'s mass m_u among its virtual objects $\boldsymbol{o}_v \xleftarrow{\boldsymbol{o}_p} \boldsymbol{o}_u$ with

$$\text{(2.13)} \quad m_v \coloneqq m_u \cdot \frac{m_p \cdot \exp\left(\frac{\operatorname{dist}(\boldsymbol{o}_u,\boldsymbol{o}_p)^2}{-2\cdot\omega}\right)}{\sum_{\boldsymbol{o}_q \in \mathcal{O}_u^{\leftarrow}} \left(m_q \cdot \exp\left(\frac{\operatorname{dist}(\boldsymbol{o}_u,\boldsymbol{o}_q)^2}{-2\cdot\omega}\right)\right)} \ .$$

In Fig. 4c the effect of including the mass of the projector object can be observed. In every case (Eq. (2.6), (2.12) and (2.13)), the multivariate integral for each canvas Warp-kernel over the feature space $\mathbb{R}^F$ equals the mass of the data object, formally $\int_{\boldsymbol{x}\in\mathbb{R}^F} \mathrm{W}_{\mathrm{canv},i}(\boldsymbol{x})\ \mathrm{d}\boldsymbol{x} = m_i$.

Our assumptions which motivated the expansions Eq. (2.12) and (2.13) are validated by our experimental results, demonstrated in Fig. 8 and 9.

3 Approximating Warp-Kernels

In Sec. 2 we showed how real undetermined objects can be substituted by virtual determined objects. A disadvantage of the approach is that a large number of virtual determined objects is created, increasing the runtime of calculating the KDE by Eq. (2.9). We show this experimentally in Fig. 11. In the following we present two approaches to approximate the methods from Sec. 2.

3.1 Approximation via Clustering.

A straightforward approach in reducing the number of virtual determined objects $\boldsymbol{o}_d \in \mathcal{O}_u^{\leftarrow}$ is to cluster them. We use the K-Means derivative G-Means [10], where we do not have to specify the number of clusters beforehand. Each centroid of the final clustering can be considered to be a center of a Gaussian and is used as a substitute for the virtual determined objects $\boldsymbol{o}_d$ within its Voronoi cell. The mass of the objects within each cell are added and allocated to the respective substitute. After that, the substitutes are used as if they where the actual $\boldsymbol{o}_d$ and Eq. (2.9), is calculated.

3.2 Tube Warp-kernel.

As an alternative to clustering the objects in $\mathcal{O}_u^{\leftarrow}$, we approximate the canvas Warp-kernel to be uniform within $\mathbb{V}_u$. We combine the Gaussian kernel for the dimensions $f \in \mathcal{A}_i$ and the uniform kernel for the dimensions $f \in \mathcal{M}_i$. Fig. 4d visualizes such a kernel for a one dimensional $\mathbb{V}_u$ in a two dimensional feature space $\mathbb{R}^F$. Fig. 1 visualizes such kernels for one dimensional $\mathbb{V}_u$s in a three dimensional $\mathbb{R}^F$ (which look like tubes, hence the name) and a two dimensional $\mathbb{R}^F$. Length and color of the lines represent the length of the gradients at the positions marked with dots. With more missing values the corresponding tube Warp-kernels look like planes, cubes, etc.

Intuitively, the tube Warp-kernel $\mathrm{W}_{\mathrm{tube},i}$ considers missing values to have never existed – compared to values that got lost. Thus, the task of the kernel is to only use those features of an object that add information, ignoring dimensions of missing features. Now, we can formalize $\mathrm{W}_{\mathrm{tube},i}$ and $\mathrm{WDF}_{\mathrm{tube}}$ as

$$\text{(3.14)} \quad \begin{aligned} \mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x}) &\coloneqq \frac{m_i \cdot \exp(\delta_{\mathrm{tube},i}(\boldsymbol{x})\ /\ (-2\cdot\omega))}{\varphi_i \cdot \varepsilon_i \cdot \sqrt{(2\cdot\pi\cdot\omega)^{|\mathcal{A}_i|}}} \ , \\ \delta_{\mathrm{tube},i}(\boldsymbol{x}) &\coloneqq \sum_{f\in\mathcal{A}_i} \left(\ x_{:f} - o_{i:f}\ \right)^2 \ , \end{aligned}$$

$$\text{(3.15)} \quad \mathrm{WDF}_{\mathrm{tube}}(\boldsymbol{x}) \coloneqq \sum_{i=1}^{|\mathcal{O}|} \mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x})$$

where $\varphi_i \in \mathbb{R}_{>0}$ and $\varepsilon_i \in \mathbb{R}_{>0}$ are strictly positive compensation parameters. Note that in Eq. (3.14) we assume that we are only interested in the density estimation within a data space $\mathbb{D} \subset \mathbb{R}^F$, a compact space within the feature space. $\mathbb{D}$, φ_i, and ε_i will be explained in the following. It is worth noting that every object might have missing values in different dimensions and a different $\varphi_i \cdot \varepsilon_i$. So, each object is associated with an individual kernel function, unlike classical KDE, where each object uses the same kernel.

Data Space: Since the Warp-kernel $\mathrm{W}_{\mathrm{tube},i}$ represents the distribution of the information mass m_i (see Sec. 2.2), we want the integral of $\mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x})$ over a – yet to be defined – data space $\mathbb{D}$ to equal m_i for every data object $\boldsymbol{o}_i$, formally

$$(3.16)\quad \int_{\boldsymbol{x}\in\mathbb{D}} \mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x})\ \mathrm{d}\boldsymbol{x} \overset{!}{=} m_i\ , \forall i \in \{1, ..., |\mathcal{O}|\}\ .$$

In order to fulfill this requirement, $\varphi_i \cdot \varepsilon_i$ is used. We are going to define φ_i and ε_i on the basis of $\mathbb{D}$.

In classical KDE the data space $\mathbb{D}$ equals the feature space $\mathbb{R}^F$. This is not possible here, because fulfilling Eq. (3.16) would require $\varphi_i \cdot \varepsilon_i$ to be infinite for every undetermined object $\boldsymbol{o}_u$, reducing $\mathrm{W}_{\mathrm{tube},i}$ to absurdity.

Moreover, using a different $\varphi_i \cdot \varepsilon_i$ for each data object and reducing the data space to be $\mathbb{D} \subset \mathbb{R}^F$ also has the advantage of improving the density model for practical applications: In these, most of the time we are interested in the density in that subset of the feature space $\mathbb{R}^F$ that is within the boundaries of the data objects. In other words, we are interpolating the density, not extrapolating. The closer a data object is to the multivariate mean of all feature values (ignoring missing values), the higher its degree of influence on the density in that subset of interest. The location-sensitive φ_i of $\mathrm{W}_{\mathrm{tube},i}$ counteracts this effect (details follow, see also Eq. (3.19)). Considering this we define data space as

$$(3.17)\quad \mathbb{D} \coloneqq \left\{ \boldsymbol{x} \in \mathbb{R}^F \mid \varGamma_{\min:f} \leqslant x_{:f} \leqslant \varGamma_{\max:f}\ , \forall\ f \in \mathcal{F} \right\}\ .$$

A useful definition of $\boldsymbol{\varGamma}_{\min} \in \mathbb{R}^F$, $\boldsymbol{\varGamma}_{\max} \in \mathbb{R}^F$ is $\varGamma_{\min:f} = \min(\{o_{i:f} \mid o_{i:f} \neq \boxslash\})$, $\varGamma_{\max:f} = \max(\{o_{i:f} \mid o_{i:f} \neq \boxslash\})$. Depending on the situation, it might make sense to increase $\mathbb{D}$. Without loss of generality, we assume $\varGamma_{\min:f} \neq \boxslash$, $\varGamma_{\max:f} \neq \boxslash$ for all $f \in \mathcal{F}$. $\mathbb{D}$ is showcased in Fig. 3 where the PDF is displayed that is created by the three determined and three undetermined objects displayed in Fig. 2.

Object-adaptive normalization parameters: Since $\mathrm{W}_{\mathrm{tube},i}$ represents the distribution of the information mass m_i, the integral of $\mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x})$ over $\mathbb{D}$ should equal m_i as in Eq. (3.16). Due to the varying number of missing values per object, this can only be realized by an *object-adaptive* normalization

$$(3.18)\qquad \varepsilon_i \coloneqq \prod\nolimits_{f\in\mathcal{M}_i} \varGamma_{\max:f} - \varGamma_{\min:f}\ .$$

The effect of ε_i can be easily seen in Fig. 4d, where the kernel of the determined object is considerably higher than the kernel for the undetermined object.

With ε_i we adjust $\mathrm{W}_{\mathrm{tube},i}$ according to the concept that a data object influences the WDF in and only in $\mathbb{D}$. However ε_i adjusts for $\boldsymbol{o}_i$'s incomplete features only. This means that an object's $\boldsymbol{o}_i$ incomplete features are handled as if they could only take on values from $\mathbb{D} \cap \mathbb{V}_i$, while each complete feature is considered to originate from $\mathbb{R}$. This is rectified with

$$(3.19)\quad \varphi_i \coloneqq \int_{\boldsymbol{x}\in\mathbb{D}\ \mid\ \boldsymbol{x}^{\intercal}\cdot(1-\boldsymbol{e}_i)=0} \frac{\exp\left(\frac{\|\ \boldsymbol{e}_i\circ\boldsymbol{x}-\boldsymbol{e}_i\circ\boldsymbol{o}_i)\ \|_2^2}{-2\cdot\omega}\right)}{\sqrt{(2\cdot\pi\cdot\omega)^{|\mathcal{A}_i|}}}\ \mathrm{d}\boldsymbol{x}\ .$$

which is the integral of the marginal Gaussian over the data space in the dimensions of the available features. The result of this normalization is twofold: the condition Eq. (3.16) holds, and $\mathrm{W}_{\mathrm{tube},i}(\boldsymbol{x})$ becomes location-sensitive. Fig. 5 and 6 display the effect of location-sensitivity on a complete dataset, showing that the φ_i balances $\mathrm{WDF}_{\mathrm{tube}}$ in $\mathbb{D}$.

To summarize: We introduced three new kernels for undetermined objects – and approximations either via clustering or the tube Warp-kernel. Each of the kernels is object dependent, considering the varying number of missing features and the location in space. This concept is in strong contrast to classical KDE, where each object uses the same kernel.

4 Experiments

In the following we present the results of two types of experiments: First, large-scale experiments which quantify the robustness of density estimation with Warp-kernels against missing values. We define robustness as the ability to change as little as possible in results in the face of increasing difficulty. Here, "difficulty" is the number of missing values and "result" the estimated density, which should change as little as possible compared to the complete datasets. Thus, the density of the complete dataset is the ground truth. Second, in Sec. 4.9, we show a data mining pipeline scenario in which using the tube Warp-kernel is superior to using imputation.

Our experiments compare in three ways: First we compare the canvas Warp-KDEs with each other. Then we confirm that the tube Warp-KDE is a fast and accurate approximation contrary to the approximation via clustering. Finally we compare those methods to pre-processing with imputation, demonstrating that the tube Warp-KDE is clearly more accurate.

4.1 Datasets. For the experiments we chose the five 2-dimensional artificial datasets "aggregation" [7] (788 objects), "pathbased" [3] (300 obj.), "R15" [21] (600 obj.), "jain" [13] (373 obj.), and "flame" [6] (240 obj.). For each of these datasets we sequentially deleted values, first until 5% were missing, then further until 10%, then until

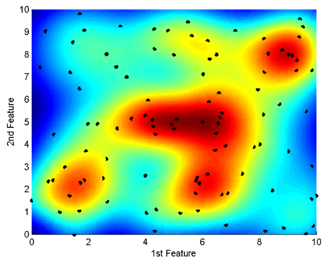


Fig. 5. Loc.-insens. $\mathrm{WDF}_{\mathrm{cpl}}$

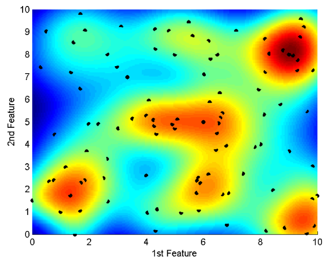


Fig. 6. Loc.-sens. $\mathrm{WDF}_{\mathrm{tube}}$

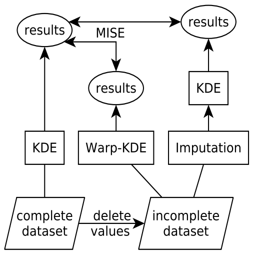


Fig. 7. Setup

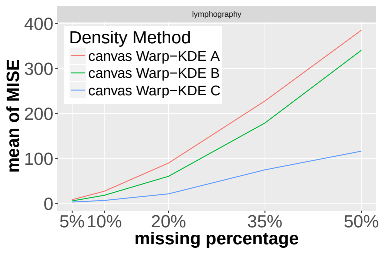


Fig. 8. Density estimation

20%, then until 35%, finally until 50% were missing, i.e. highly incomplete data. Since values were deleted randomly, we repeated the process up to 19 times for each of the five datasets, resulting in 475 datasets.

4.2 Setup. To measure the performance of density methods, the Mean Integrated Squared Error (MISE)

$$MISE = \sum (\mathrm{WDF}_{true}(\boldsymbol{x}) - \mathrm{WDF}_{estim.}(x))^2 \tag{4.20}$$

is a popular choice. With the true density unavailable, many publications cannot compute the MISE and calculate the Asymptotic MISE (AMISE) instead. Aiming for robustness and having the ground truth available, we do not need to fall back on the AMISE and calculated the MISE directly by

$$MISE = \sum (\mathrm{WDF}_{cpl.}(\boldsymbol{x}) - \mathrm{WDF}_{incpl.}(x))^2\ . \tag{4.21}$$

The concept of our experiments is illustrated in Fig. 7: First, we ran KDE on the complete dataset and computed the densities for about 900 sample points on a grid within the data space $\mathbb{D}$ as the reference result. Then, on the one hand, we ran Warp-KDE on the incomplete datasets, and on the other hand, imputed the missing values before applying KDE.

We ran experiments for ten methods: Three canvas Warp-KDE, three canvas Warp-KDE approximated via clustering (which results are too similar to be differentiated), tube Warp-KDE, three pipelines with imputation pre-processing. To ensure comparability, in every case the same sample points were used and all kernels were made location sensitive with φ_i. The bandwidth was selected by hand for each complete dataset and not changed over the course of the experiments. In total, more than 11 400 experimental runs were performed.

4.3 Warp-KDE based on Virtual Objects. First, we ran the experiments for the methods based on virtual objects presented in Sec. 2. We denote the results using Eq. (2.6) with A, results using Eq. (2.12) with B and results using Eq. (2.13) with C. In Fig. 9 we see that, per dataset, the results do not vary much from each other. Unsurprisingly, the MISE varies from dataset to dataset. B and C have the same results because the datasets are two dimensional: There is only one iteration of derivation in Algorithm 1, so all projectors are real with the same mass, leading to identical virtual objects.

Fig. 8 shows the results of an additional experiment with the 3D lymphography dataset (148 obj.). We see that there is indeed a difference for the methods Eq. (2.12) and (2.13). This dataset is a good example for a usage of different masses in real objects: We can reduce the number of the real objects substantially by aggregating those with the same location. On such a location we place a substitute object whose mass equals the sum of the aggregated objects. In the following, the substitute objects are used instead of the original ones.

4.4 Approximations via Clustering. As established, the number of determined objects $|\mathcal{O}_{\mathrm{d}}|$ required in Eq. (2.9) increases very fast with the number of missing values. Thus, we introduced an approximation via clustering in Sec. 3.1. In Fig. 9 we compare the virtual object Warp-KDEs from Sec. 2 with the approximation via clustering. We notice that, despite adding quite a bit of algorithmic complexity, the line representing the MISE is substantially higher and thus worse than the original Warp-KDEs – independent of the dataset. The reason is that G-Means clustering has trouble to find Gaussian distributed Voronoi cells. This approach does not lead to desired results.

4.5 Tube Warp-KDE. Fig. 9 also shows the results of the tube Warp-KDE approximation from Sec. 3.2. As expected, in general the canvas Warp-KDEs have a lower MISE than the tube Warp-kernel approximation, only for the “flame” dataset the tube Warp-KDE is surprisingly strong. However, in contrast to the approximation via clustering, we notice that the results are very close to the original Warp-KDEs. Additionally, the tube Warp-KDE is much faster than any other previous method. In fact, the speed is independent of the number of missing values, since the number of kernels stays the same in Eq. (3.15). Fig. 11 displays the runtime for the “flame” dataset: Note that the tube Warp-KDE is the red line at the bottom of the figure.

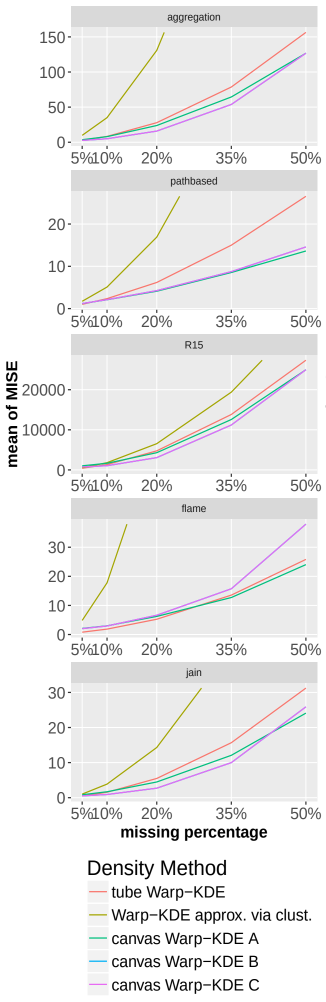


Fig. 9. Density estimation

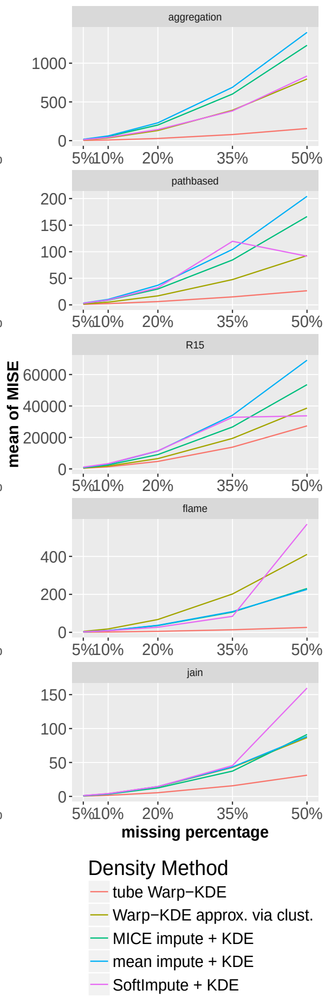


Fig. 10. Density estimation

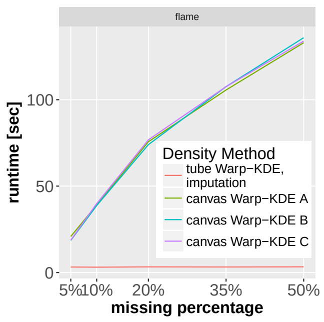


Fig. 11. Runtime for "flame"

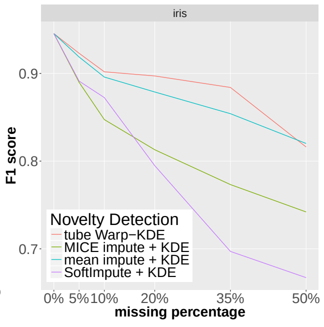


Fig. 12. Novelty detection

4.6 Pre-Processing with Imputations. Without having an integrative method like Warp-KDE, a typical Data Science approach would be to impute the missing values in a pre-processing step. Thus we compare the tube Warp-KDE with three different imputation methods: SimpleFill (a missing value is imputed with its feature's mean), Soft Impute [16] and MICE [1]. In Fig. 10 we can observe that the MICE and mean impute are somewhat similar, with MICE being slightly better, while SoftImpute's results are hard to predict results. In every case, except MICE and mean impute for the "flame" dataset, the KDE with pre-processed imputation is even worse than for the approximation via clustering. Clearly, the tube Warp-KDE performs better than imputation.

4.7 Variance. The tube Warp-KDE is not only a good approximation of the canvas Warp-KDEs with a very low computational cost, it has very low variance across the datasets as well. Its variance is on par with the original methods, or even better, as exemplified for the "flame" dataset in Tab. 1.

4.8 Recommendation. From the presented experiments we can clearly see that while the canvas Warp-KDEs are very effective, they are also very slow. While the approximation by clustering and pre-processing with imputation are too ineffective, tube Warp-KDE is faster than imputation and nearly as effective as the canvas Warp-KDEs. Thus we recommend to use tube Warp-KDE in real-life applications.

4.9 Robustness in a Data Mining Pipeline. It is not uncommon to use density estimation as a component in data mining pipelines. In this experiment we show a case in which using a tube Warp-kernel is superior to using an imputation pre-processing component. Since we established already in the previous experiments that the canvas Warp-KDE methods are too slow for real-life applications and approximation via clustering too ineffective, we concentrate on tube Warp-KDE and the imputation approaches. For that we chose the task of novelty detection on the famous iris dataset (150 obj.).

Setup: In three setups we chose each iris type as abnormal and the other two types as normal. We sampled 90% of each normal type into the training dataset and the remaining 10% into the test dataset. Additionally, we sampled 20% of the abnormal type into the test dataset. For each setup we sequentially deleted training values until 5% were missing, then until 10%, 20%, 35%, 50% were missing. Thus, including 0% missing, we had 18 experimental scenarios.

Results: Fig. 12 displays the mean of the F1 scores of the three setups, demonstrating that the pipeline using the tube Warp-kernel has better results than any imputation method for nearly all missing percentages. However, we do not claim that using Warp-kernels is the way to go for novelty detection in general, since it was not primarily developed for it, and further large-scale experiments would need to be conducted.

Table 1. Variances for “flame” data, best values in green.

	method	Var. of MISE for miss. % 5%	10%	20%	35%	50%
vir. obj.	canvas Warp-KDE A	2.1	3.0	6.2	12.7	24.0
	canvas Warp-KDE B	2.0	2.9	6.6	15.7	37.9
	canvas Warp-KDE C	2.0	2.9	6.6	15.7	37.9
appr.	approx. via clust.	4.8	17.8	67.6	201.9	410.5
	tube Warp-KDE	0.8	1.9	5.2	13.5	25.8
impute	MICE impute + KDE	2.5	8.9	35.9	106.5	230.8
	mean impute + KDE	2.6	9.2	36.9	109.3	226.1
	SoftImpute + KDE	3.0	8.4	26.9	84.3	569.8

5 Conclusion

This paper introduces multiple novel kernels for density estimation that can handle missing values, even if no feature is complete. The presented Warp-kernels fulfill these objectives without the need of imputation or marginalization. By constructing virtual objects, the density within the missing features of an objects are intuitively modeled via KDE as well. Our experiments show that the canvas Warp-KDEs with virtual objects deliver the best results, but to very high computational cost. Thus, approximating approaches were introduced. We showed that a naive approximation via clustering is not sufficient, and introduced the tube Warp-kernel. The tube Warp-kernel worsens the MISE only slightly, while the computational cost stays constant – independent of the number of missing values. Experimental results demonstrate that the novel kernel density estimation generalization is superior to imputing missing values, even if datasets are highly incomplete.

Acknowledgments. The authors thank Konrad Leibrandt from the Department of Computing, Imperial College London for providing valuable suggestions for the distributed computation of the experiments.

This research was supported by the German Research Foundation, Emmy Noether grant GU 1409/2-1, and by the Technical University of Munich - Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement no 291763, co-funded by the European Union.

References

[1] M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf, *Multiple imputation by chained equations: what is it and how does it work?*, Int. Journ. of Methods in Psychiatr. Res., 20 (2011), pp. 40–49.

[2] M. Á. Carreira-Perpiñán and C. Williams, *On the number of modes of a gaussian mixture*, report, Inst. for Adapt. and Neur. Comp., Univ. of Edinburgh, 2003.

[3] H. Chang and D.-Y. Yeung, *Robust path-based spectral clustering with application to image segmentation*, in IEEE ICCV, 2005, pp. 278–285.

[4] S. R. Dubnicka, *Kernel density estimation with missing data and auxiliary variables*, Australian & New Zealand Journ. of Statistics, 51 (2009), pp. 247–270.

[5] R. P. D. Freedman and R. Purves, *Statistics*, Norton, New York, 1998.

[6] L. Fu and E. Medico, *FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data*, BMC Bioinform., 8 (2007), p. 3.

[7] A. Gionis, H. Mannila, and P. Tsaparas, *Clustering aggregation*, ACM TKDD, 1 (2007), pp. 1–30.

[8] S. Günnemann, I. Färber, and T. Seidl, *Multi-view clustering using mixture models in subspace projections*, in KDD, 2012, pp. 132–140.

[9] S. Günnemann, E. Müller, S. Raubach, and T. Seidl, *Flexible fault tolerant subspace clustering for data with missing values*, in IEEE ICDM, 2011, pp. 231–240.

[10] G. Hamerly and C. Elkan, *Learning the k in k-means*, in NIPS, vol. 17, 2003.

[11] R. J. Hathaway and J. C. Bezdek, *Fuzzy c-means clustering of incomplete data*, IEEE Cybernetics, 31 (2001), pp. 735–744.

[12] L. Himmelspach and S. Conrad, *Clustering approaches for data with missing values: Comparison and evaluation*, in 5th ICDIM, 2010.

[13] A. K. Jain and M. H. C. Law, *Data clustering: A user's dilemma*, in 1st Int. C. PReMI, vol. 3776 of LNCS, 2005, pp. 1–10.

[14] V. Kobayashi, T. Aluja, and L. Belanche, *Handling missing values in kernel methods with application to microbiology data*, in ESANN, 2013.

[15] R. J. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, John Wiley & Sons, 1987.

[16] R. Mazumder, T. Hastie, and R. Tibshirani, *Spectral regularization algorithms for learning large incomplete matrices*, JMLR, 11 (2010), pp. 2287–2322.

[17] G. Nebot-Troyano and L. A. B. Muñoz, *A kernel extension to handle missing data*, in SGAI Int. Conf. on Artificial Intelligence, Springer, 2009, pp. 165–178.

[18] H. Timm, C. Döring, and R. Kruse, *Fuzzy Cluster Analysis of Partially Missing Data Sets*, 2nd Int. W. on Hybr. Meth. for Adap. Sys. I, (2002), pp. 426–431.

[19] D. M. Titterington and G. M. Mill, *Kernel-based density estimates from incomplete data*, J. R. Stat. Soc. Ser. B. Methodological, 45 (1983), pp. 258–266.

[20] O. Troyanskaya et al., *Missing value estimation methods for dna microarrays*, Bioinform., 17 (2001), pp. 520–525.

[21] C. J. Veenman, M. J. T. Reinders, and E. Backer, *A maximum variance cluster algorithm*, IEEE TPAMI, 24 (2002), pp. 1273–1280.

[22] K. L. Wagstaff, *Clustering with missing values: No imputation required*, in IFCS, 2004, pp. 649–658.