ROME: Robust Multi-Modal Density Estimator

Mészáros, A.; Schumann, J.F.; Alonso-Mora, J.; Zgonnikov, A.; Kober, J.

DOI
10.24963/ijcai.2024/525

Publication date
2024

Document Version
Final published version

Published in
Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence

ROME: Robust Multi-Modal Density Estimator*

Anna Mészáros , **Julian F. Schumann** , **Javier Alonso-Mora** ,
Arkady Zgonnikov and **Jens Kober**
Cognitive Robotics, TU Delft, Netherlands
{A.Meszaros, J.F.Schumann, J.AlonsoMora, A.Zgonnikov, J.Kober}@tudelft.nl

Abstract

The estimation of probability density functions is a fundamental problem in science and engineering. However, common methods such as kernel density estimation (KDE) have been demonstrated to lack robustness, while more complex methods have not been evaluated in multi-modal estimation problems. In this paper, we present ROME (ROBust Multi-modal Estimator), a non-parametric approach for density estimation which addresses the challenge of estimating multi-modal, non-normal, and highly correlated distributions. ROME utilizes clustering to segment a multi-modal set of samples into multiple uni-modal ones and then combines simple KDE estimates obtained for individual clusters in a single multi-modal estimate. We compared our approach to state-of-the-art methods for density estimation as well as ablations of ROME, showing that it not only outperforms established methods but is also more robust to a variety of distributions. Our results demonstrate that ROME can overcome the issues of over-fitting and over-smoothing exhibited by other estimators.

1 Introduction

Numerous processes are non-deterministic by nature, from geological and meteorological occurrences, biological activities, as well as the behavior of living beings. Estimating the underlying probability density functions (PDFs) of such processes enables a better understanding of them and opens possibilities for probabilistic inference regarding future developments. Density estimation is instrumental in many applications including classification, anomaly detection, and evaluating probabilistic AI models, such as generative adversarial networks [Goodfellow *et al.*, 2020], variational autoencoders [Wei *et al.*, 2020], normalizing flows [Kobyzev *et al.*, 2020], and their numerous variations.

When probabilistic models are trained on multi-modal data, they are often evaluated using simplistic metrics (e.g., mean squared error (MSE) between the predicted and ground truth samples). However, such simplistic metrics are unsuited for determining how well a predicted distribution corresponds to the underlying distribution, as they do not capture the fit to the whole distribution. For example, the lowest MSE value between true and predicted samples could be achieved by accurate predictions of the mean of the true underlying distribution whereas potential differences in variance or shape of the distribution would not be penalized. This necessitates more advanced metrics that evaluate the match between the model and the (potentially multi-modal) data. For instance, negative log-likelihood (NLL), Jensen-Shannon divergence (JSD), and expected calibration error (ECE) can be used to evaluate how well the *full distribution* of the data is captured by the learned models [Xu *et al.*, 2019; Mozaffari *et al.*, 2020; Rasouli, 2020]. However, most data-driven models represent the learned distribution implicitly, only providing individual samples and not the full distribution as an output. This complicates the comparison of the model output to the ground-truth data distributions since the above metrics require distributions, not samples, as an input. Practically, this can be addressed by estimating the predicted probability density based on samples generated by the model.

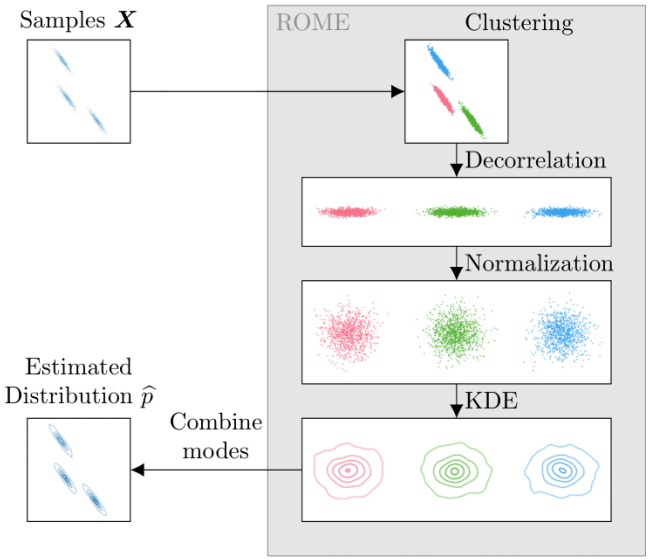

Figure 1: ROME takes samples from unknown distributions and estimates their densities to enable further downstream applications.

*Extended version with appendix: https://arxiv.org/abs/2401.10566

Simple methods like Gaussian mixture models (GMM), kernel density estimation (KDE) [Deisenroth *et al.*, 2020], and k-nearest neighbors (kNN) [Loftsgaarden and Quesenberry, 1965] are commonly used for estimating probability density functions. These estimators however rely on strong assumptions about the underlying distribution, and can thereby introduce biases or inefficiencies in the estimation process. For example, problems can arise when encountering multi-modal, non-normal, and highly correlated distributions (see Section 2). While more advanced methods such as manifold Parzen windows (MPW) [Vincent and Bengio, 2002] and vine copulas (VC) [Nagler and Czado, 2016] exist, they have not been thoroughly tested on such problems, which raises questions about their performance.

To overcome these limitations, we propose a novel density estimation approach: RObust Multi-modal Estimator (ROME). ROME employs a non-parametric clustering approach to segment potentially multi-modal distributions into separate uni-modal ones (Figure 1). These uni-modal sub-distributions are then estimated using a downstream probability density estimator (such as KDE). We test our proposed approach against a number of existing density estimators in three simple two-dimensional benchmarks designed to evaluate a model's ability to successfully reproduce multi-modal, highly-correlated, and non-normal distributions. Finally, we test our approach in a real-world setting using a distribution over future trajectories of human pedestrians created based on the Forking Paths dataset [Liang *et al.*, 2020].

2 Related Work

The most common class of density estimators are so-called Parzen windows [Parzen, 1962], which estimate the density through an aggregation of parametric probability density functions. A number of common methods use this approach, with KDE being a common non-parametric method [Silverman, 1998]. Provided a type of kernel – which is oftentimes a Gaussian but can be any other type of distribution – KDE places the kernels around each sample of a data distribution and then sums over these kernels to get the final density estimation over the data. This method is often chosen as it does not assume any underlying distribution type [Silverman, 1998]. However, if the underlying distribution is highly correlated, then the common use of a single isotropic kernel function can lead to over-smoothing [Vincent and Bengio, 2002; Wang *et al.*, 2009]. Among multiple approaches for overcoming this issue [Wang *et al.*, 2009; Gao *et al.*, 2022], especially noteworthy is the MPW approach [Vincent and Bengio, 2002]. It uses a unique anisotropic kernel for every datapoint, estimated based on the correlation of the k-nearest neighbors of each sample. However, it has not been previously tested in high-dimensional benchmarks, which is especially problematic as the required memory scales quadratically with the dimensionality of the problem.

Another common subtype of Parzen windows are GMMs [Deisenroth *et al.*, 2020], which assume that the data distribution can be captured through a weighted sum of Gaussian distributions. The parameters of the Gaussian distributions – also referred to as components – are estimated through expected likelihood maximization. Nonetheless, especially for non-normal distributions, one needs to have prior knowledge of the expected number of components to achieve a good fit without over-fitting to the training data or over-smoothing [McLachlan and Rathnayake, 2014].

Besides different types of Parzen windows, a prominent alternative is kNN [Loftsgaarden and Quesenberry, 1965] which uses the local density of the k nearest neighbors of every data point to estimate the overall density. While this method is non-parametric, it cannot be guaranteed that the resulting distribution will be normalized [Silverman, 1998]. This could be rectified by using Monte Carlo sampling to obtain a good coverage of the function's domain and obtain an accurate estimate of the normalization factor, which, however, becomes computationally intractable for high-dimensional distributions.

When it comes to estimating densities characterized by correlations between dimensions, copula-based methods are an often favored approach. Copula-based methods decompose a distribution into its marginals and an additional function, called a copula, which describes the dependence between these marginals over a marginally uniformly distributed space. The downside of most copula-based approaches is that they rely on choosing an adequate copula function (e.g., Gaussian, Student, or Gumbel) and estimating their respective parameters [Joe, 2014]. One non-parametric copula-based density estimator [Bakam and Pommeret, 2023] aims to address this limitation by estimating copulas with the help of superimposed Legendre polynomials. While this can achieve good results in estimating the density function, it may become computationally intractable as the distribution's dimensionality increases. Another approach involves the use of VC [Nagler and Czado, 2016], which assume that the whole distribution can be described as a product of bivariate Gaussian distributions, thus alleviating the curse of dimensionality. Its convergence, however, can only be guaranteed for normal distributions. Elsewhere [Otneim and Tjøstheim, 2017], a similar approach was pursued, with changes such as using logarithmic splines instead of univariate KDEs for estimating the marginals. However, both of these approaches are not designed for multi-modal distributions and have not been thoroughly tested on such problems.

3 RObust Multi-modal Estimator (ROME)

The problem of density estimation can be formalized as finding a queryable $\widehat{p} \in \mathcal{P}$, where

$$\mathcal{P} = \left\{ g : \mathbb{R}^M \to \mathbb{R}^+ \mid \int g(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} = 1 \right\},$$

such that $\widehat{p}$ is close to the non-queryable PDF p underlying the N available M-dimensional samples $\boldsymbol{X} \in \mathbb{R}^{N \times M}$: $\boldsymbol{X} \sim p$.

A solution to the above problem would be an estimator $f : \mathbb{R}^{N \times M} \to \mathcal{P}$, resulting in $\widehat{p} = f(\boldsymbol{X})$. Our proposed estimator f_{ROME}[1] (Algorithm 1) is built on top of non-parametric cluster extraction. Namely, by separating groups of samples surrounded by areas of low density – expressing the mode of the underlying distribution – we reduce the multi-modal

[1]Source code: https://github.com/anna-meszaros/ROME

Algorithm 1 ROME

function TRAINROME($\boldsymbol{X}$)
  ▷ *Clustering (OPTICS)*
  $\boldsymbol{X}_{I,N}, \boldsymbol{R}_N \leftarrow$ REACHABILTYANALYSIS($\boldsymbol{X}$)
  $\boldsymbol{C}, S \leftarrow \{\{1,\dots,N\}\}, -1.1$
  for all $\epsilon \in \boldsymbol{\varepsilon}$ **do**
    $\boldsymbol{C}_\epsilon \leftarrow$ **DBSCAN**$(\boldsymbol{R}_{I,N}, \epsilon)$
    $S_\epsilon \leftarrow$ **SIL**$(\boldsymbol{C}_\epsilon, \boldsymbol{X}_{I,N})$
    if $S_\epsilon > S$ **then**
      $\boldsymbol{C}, S \leftarrow \boldsymbol{C}_\epsilon, S_\epsilon$
  for all $\xi \in \boldsymbol{\xi}$ **do**
    $\boldsymbol{C}_\xi \leftarrow$ **$\boldsymbol{\xi}$-clustering**$(\boldsymbol{R}_{I,N}, \xi)$
    $S_\xi \leftarrow$ **SIL**$(\boldsymbol{C}_\xi, \boldsymbol{X}_{I,N})$
    if $S_\xi > S$ **then**
      $\boldsymbol{C}, S \leftarrow \boldsymbol{C}_\xi, S_\xi$
  for all $C \in \boldsymbol{C}$ **do**
    ▷ *Decorrelation*
    $\overline{\boldsymbol{x}}_C \leftarrow \text{MEAN}(\boldsymbol{X}_C)$
    $\overline{\boldsymbol{X}}_C \leftarrow \boldsymbol{X}_C - \overline{\boldsymbol{x}}_C$
    $\boldsymbol{R}_C \leftarrow PCA(\overline{\boldsymbol{X}}_C)$
    ▷ *Normalization*
    $\widetilde{\boldsymbol{\Sigma}}_C \leftarrow \text{STD}(\overline{\boldsymbol{X}}_C \boldsymbol{R}_C^T)$
    $\boldsymbol{T}_C \leftarrow \boldsymbol{R}_C^T \widetilde{\boldsymbol{\Sigma}}_C^{-1}$
    ▷ *PDF Estimation*
    $\widehat{p}_C \leftarrow f_{\text{KDE}}\left(\overline{\boldsymbol{X}}_C \boldsymbol{T}_C\right)$
  return $\boldsymbol{C}, \{\widehat{p}_C, \overline{\boldsymbol{x}}_C, \boldsymbol{T}_C \,|\, C \in \boldsymbol{C}\}$

function QUERYROME($\boldsymbol{x}$, $\boldsymbol{X}$)
  $\boldsymbol{C}, \{\widehat{p}_C, \overline{\boldsymbol{x}}_C, \boldsymbol{T}_C \,|\, C \in \boldsymbol{C}\} \leftarrow$ TRAINROME($\boldsymbol{X}$)
  $l = \boldsymbol{0}$
  for all $C \in \boldsymbol{C}$ **do**
    $\widehat{\boldsymbol{x}} \leftarrow (\boldsymbol{x} - \overline{\boldsymbol{x}}_C)\, \boldsymbol{T}_C$
    $l \leftarrow l + \ln(\widehat{p}_C(\widehat{\boldsymbol{x}})) + \ln(|C|) - \ln N + \ln(|\det(\boldsymbol{T}_C)|)$
  return $\exp(l)$

density estimation problem to multiple uni-modal density estimation problems for each cluster. The distributions within each cluster then become less varied in density or correlation than the full distribution. Combining this with decorrelation and normalization, the use of established methods such as KDE to estimate probability densities for those uni-modal distributions is now more promising, as problems with multi-modality and correlated modes (see Section 2) are accounted for. The multi-modal distribution is then obtained as a weighted average of the estimated uni-modal distributions.

3.1 Extracting Clusters

To cluster samples $\boldsymbol{X}$, ROME employs the OPTICS algorithm [Ankerst *et al.*, 1999] that can detect clusters of any shape with varying density using a combination of reachability analysis – which orders the data in accordance to reachability distances – followed by clustering the ordered data based on these reachability distances.

In the first part of the algorithm, the reachability analysis is used to sequentially transfer samples from a set of unincluded samples $\boldsymbol{X}_{U,i}$ to the set of included and ordered samples $\boldsymbol{X}_{I,i}$, starting with a random sample $\boldsymbol{x}_1$ ($\boldsymbol{X}_{I,1} = \{\boldsymbol{x}_1\}$ and $\boldsymbol{X}_{U,1} = \boldsymbol{X} \backslash \{\boldsymbol{x}_1\}$). The samples $\boldsymbol{x}_{i+1}$ are then selected at iteration i based on the reachability distance r:

$$\boldsymbol{x}_{i+1} = \underset{\boldsymbol{x} \in \boldsymbol{X}_{U,i}}{\arg\min}\, r(\boldsymbol{x}, \boldsymbol{X}_{I,i}) = \underset{\boldsymbol{x} \in \boldsymbol{X}_{U,i}}{\arg\min} \; \underset{\widetilde{\boldsymbol{x}} \in \boldsymbol{X}_{I,i}}{\min} \; d_r\left(\boldsymbol{x}, \widetilde{\boldsymbol{x}}\right) . \tag{1}$$

This sample is then transferred between sets, with $\boldsymbol{X}_{I,i+1} = \boldsymbol{X}_{I,i} \cup \{\boldsymbol{x}_{i+1}\}$ and $\boldsymbol{X}_{U,i+1} = \boldsymbol{X}_{U,i} \backslash \{\boldsymbol{x}_{i+1}\}$, while expanding the reachability set $\boldsymbol{R}_{i+1} = \boldsymbol{R}_i \cup \{r(\boldsymbol{x}_{i+1}, \boldsymbol{X}_{I,i})\}$ (with $\boldsymbol{R}_1 = \{\infty\}$). The reachability distance d_r in Equation (1) is defined as

$$d_r\left(\boldsymbol{x}, \widetilde{\boldsymbol{x}}\right) = \max\left\{ \|\boldsymbol{x} - \widetilde{\boldsymbol{x}}\| \,,\; \underset{\widehat{\boldsymbol{x}} \in \boldsymbol{X} \backslash \{\boldsymbol{x}\}}{\min_{k_{\boldsymbol{c}}}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\| \right\},$$

where $\min_{k_{\boldsymbol{c}}}$ is the $k_{\boldsymbol{c}}$-smallest value of all the available $\widehat{\boldsymbol{x}}$, used to smooth out random local density fluctuations. We use

$$k_{\boldsymbol{c}} = \min\left\{ k_{\max}, \max\left\{ k_{\min}, \frac{NM}{\alpha_k} \right\} \right\}, \tag{2}$$

where $k_{\min}$ and $k_{\max}$ ensure there are sufficient but not too many points for this smoothing, while the term NM/α_k adjusts $k_{\boldsymbol{c}}$ to the number of samples and dimensions.

The second part of the OPTICS algorithm – after obtaining the reachability distances $\boldsymbol{R}_N$ and the ordered set $\boldsymbol{X}_{I,N}$ – is the extraction of a set of clusters $\boldsymbol{C}$, with clusters $C = \{c_{\min}, \dots, c_{\max}\} \in \boldsymbol{C}$ (with $\boldsymbol{X}_C = \{\boldsymbol{x}_{I,N,j} \,|\, j \in C\} \in \mathbb{R}^{|C| \times M}$). As the computational cost of creating such a cluster set is negligible compared to the reachability analysis, we can test multiple clusterings generated using two different approaches (with $r_{\text{bound}} = \min\{r_{N,c_{\min}}, r_{N,c_{\max}+1}\}$, see Appendix A for further discussion):

- First, we use **DBSCAN** [Ester *et al.*, 1996] for generating the clustering $\boldsymbol{C}_\epsilon$ based on an absolute limit ϵ, where a cluster must fulfill the condition:

$$r_{N,c} < \epsilon \leq r_{\text{bound}} \; \forall\, c \in C \backslash \{c_{\min}\} . \tag{3}$$

- Second, we use **$\boldsymbol{\xi}$-clustering** [Ankerst *et al.*, 1999] to generate the clustering $\boldsymbol{C}_\xi$ based on the proportional limit ξ, where a cluster fulfills:

$$\xi \leq 1 - \frac{r_{N,c}}{r_{\text{bound}}} \; \forall\, c \in C \backslash \{c_{\min}\} . \tag{4}$$

In both cases, each prospective cluster C also has to fulfill the condition $|C| \geq 2$. However, it is possible that not every sample can be fit into a cluster fulfilling the conditions above. These samples are then kept in a separate noise cluster C_{noise} that does not have to fulfill those conditions ($C_{\text{noise}} \in \boldsymbol{C}_\epsilon$ or $C_{\text{noise}} \in \boldsymbol{C}_\xi$ respectively). Upon generating multiple sets of clusters $\boldsymbol{C}_\epsilon$ ($\epsilon \in \boldsymbol{\varepsilon}$) and $\boldsymbol{C}_\xi$ ($\xi \in \boldsymbol{\xi}$), we select the final set of clusters $\boldsymbol{C}$ that achieves the highest silhouette score[2] $S = \textbf{SIL}(\boldsymbol{C}, \boldsymbol{X}_{I,N}) \in [-1, 1]$ [Rousseeuw, 1987]. The clustering then allows us to use PDF estimation methods on uni-modal distributions.

[2]The silhouette score measures the similarity of each object to its own cluster's objects compared to the other clusters' objects.

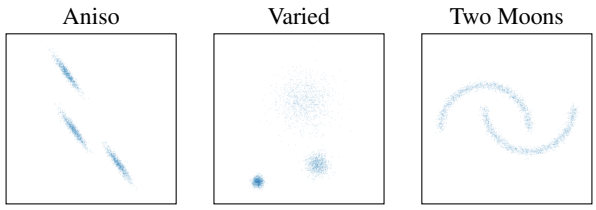

Figure 2: Samples from the two-dimensional synthetic distributions used for evaluating different PDF estimators.

3.2 Feature Decorrelation

In much of real-life data, such as the distributions of a person's movement trajectories, certain dimensions of the data are likely to be highly correlated. Therefore, the features in each cluster $C \in \boldsymbol{C}$ should be decorrelated using a rotation matrix $\boldsymbol{R}_C \in \mathbb{R}^{M\times M}$. In ROME, $\boldsymbol{R}_C$ is found using principal component analysis (PCA) [Wold *et al.*, 1987] on the cluster's centered samples $\overline{\boldsymbol{X}}_C = \boldsymbol{X}_C - \overline{\boldsymbol{x}}_C$ ($\overline{\boldsymbol{x}}_C$ is the cluster's mean value). An exception are the noise samples in C_{noise}, which are not decorrelated (i.e., $\boldsymbol{R}_{C_{\text{noise}}} = \boldsymbol{I}$). One can then get the decorrelated samples $\boldsymbol{X}_{\text{PCA},C}$:

$$\boldsymbol{X}^T_{\text{PCA},C} = \boldsymbol{R}_C \overline{\boldsymbol{X}}^T_C \,.$$

3.3 Normalization

After decorrelation, we use the matrix $\widetilde{\boldsymbol{\Sigma}}_C \in \mathbb{R}^{M\times M}$ to normalize $\boldsymbol{X}_{\text{PCA},C}$:

$$\widehat{\boldsymbol{X}}_C = \boldsymbol{X}_{\text{PCA},C}\left(\widetilde{\boldsymbol{\Sigma}}_C\right)^{-1} = \overline{\boldsymbol{X}}_C \boldsymbol{R}^T_C \left(\widetilde{\boldsymbol{\Sigma}}_C\right)^{-1} = \overline{\boldsymbol{X}}_C \boldsymbol{T}_C \,.$$

Here, $\widetilde{\boldsymbol{\Sigma}}_C$ is a diagonal matrix with the entries $\widetilde{\sigma}^{(m)}_C$. To avoid over-fitting to highly correlated distributions, we introduce a regularization with a value $\sigma_{\min}$ (similar to [Vincent and Bengio, 2002]) that is applied to the empirical standard deviation $\sigma^{(m)}_{\text{PCA},C} = \sqrt{\mathbb{V}_m\left(\boldsymbol{X}_{\text{PCA},C}\right)}$ ($\mathbb{V}_m$ is the variance along feature m) of each rotated feature m (with $\boldsymbol{C}' = \boldsymbol{C}\backslash\{C_{\text{noise}}\}$):

$$\widetilde{\sigma}^{(m)}_C = \begin{cases} \max\left\{\sum\limits_{\mathscr{C}\in\boldsymbol{C}'} \frac{\sqrt{\mathbb{V}_m(\boldsymbol{X}_{\mathscr{C}})}}{|\boldsymbol{C}|-1}, \sigma_{\min}\right\} & C = C_{\text{noise}} \\ \left(1 - \frac{\sigma_{\min}}{\max\limits_m \sigma^{(m)}_{\text{PCA},C}}\right)\sigma^{(m)}_{\text{PCA},C} + \sigma_{\min} & \text{otherwise} \end{cases} .$$

3.4 Estimating the Probability Density Function

Taking the transformed data (decorrelated and normalized), ROME fits the Gaussian KDE f_{KDE} on each separate cluster C as well as the noise samples C_{noise}. For a given bandwidth b_C for data in cluster C, this results in a partial PDF $\widehat{p}_C$.

$$\widehat{p}_C\left(\widehat{\boldsymbol{x}}\right) = f_{\text{KDE}}\left(\widehat{\boldsymbol{X}}_C\right)\left(\widehat{\boldsymbol{x}}\right) = \frac{1}{|C|}\sum_{\boldsymbol{\chi}\in\widehat{\boldsymbol{X}}_C} \mathcal{N}\left(\widehat{\boldsymbol{x}}|\boldsymbol{\chi}, b_C\boldsymbol{I}\right) .$$

The bandwidth b_C is set using Silverman's rule [Silverman, 1998]:

$$b_C = \left(\frac{M+2}{4}n_C\right)^{-\frac{1}{M+4}}, \; n_C = \begin{cases} 1 & C = C_{\text{noise}} \\ |C| & \text{else} \end{cases} .$$

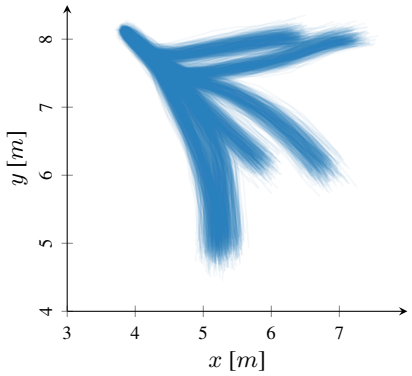

Figure 3: Samples from the multi-modal pedestrian trajectory distribution [Liang *et al.*, 2020] used for evaluating different PDF estimators. Trajectories span 12 timesteps recorded at 2.5 Hz.

To evaluate the density function $\widehat{p} = f_{\text{ROME}}(\boldsymbol{X})$ for a given sample $\boldsymbol{x}$, we take the weighted averages of each cluster's $\widehat{p}_C$:

$$\widehat{p}\left(\boldsymbol{x}\right) = \sum_{C\in\boldsymbol{C}} \frac{|C|}{N}\,\widehat{p}_C\left(\left(\boldsymbol{x} - \overline{\boldsymbol{x}}_C\right)\boldsymbol{T}_C\right)\left|\det\left(\boldsymbol{T}_C\right)\right| .$$

Here, the term $|C|/N$ is used to weigh the different distributions of each cluster with the size of each cluster, so that each sample is represented equally. As the different KDEs $\widehat{p}_C$ are fitted to the transformed samples, we apply them not to the original sample $\boldsymbol{x}$, but instead apply the identical transformation used to generate those transformed samples, using $\widehat{p}_C\left(\left(\boldsymbol{x} - \overline{\boldsymbol{x}}_C\right)\right)$. To account for the change in density within a cluster C introduced by the transformation $\boldsymbol{T}_C$, we use the factor $|\det\left(\boldsymbol{T}_C\right)|$.

4 Experiments

We compare our approach against two baselines from the literature (VC [Nagler and Czado, 2016] and MPW [Vincent and Bengio, 2002]) in four scenarios, using three metrics. Additionally, we carry out an ablation study on our proposed method ROME.

For the hyperparameters pertaining to the clustering within ROME (see Section 3.1), we found empirically that stable results can be obtained using 199 possible clusterings, 100 for DBSCAN (Equation (3))

$$\boldsymbol{\varepsilon} = \left\{\min R_N + \left(\frac{\alpha}{99}\right)^2\left(\max(R_N\backslash\{\infty\}) - \min R_N\right)\right| \\ \alpha \in \{0,\ldots,99\}\Big\}$$

combined with 99 for ξ-clustering (Equation (4))

$$\boldsymbol{\xi} = \left\{\frac{\beta}{100}\,|\,\beta\in\{1,\ldots,99\}\right\},$$

as well as using $k_{\min} = 5$, $k_{\max} = 20$, and $\alpha_k = 400$ for calculating $k_{\mathbf{c}}$ (Equation (2) and Appendix C).

4.1 Distributions

In order to evaluate different aspects of a density estimation method f, we used a number of different distributions.

Distrib.	$D_{\text{JS}} \downarrow_0^1$ ROME	MPW	VC	$\widehat{W} \to 0$ ROME	MPW	VC	$\widehat{L} \uparrow$ ROME	MPW	VC
Aniso	0.010 ± 0.001	0.026 ± 0.002	$\underline{0.005} \pm 0.001$	$\underline{-0.13} \pm 0.31$	-0.60 ± 0.13	1.91 ± 0.91	$\underline{-2.53} \pm 0.02$	-2.57 ± 0.02	-3.19 ± 0.02
Varied	0.011 ± 0.001	0.025 ± 0.002	$\underline{0.008} \pm 0.001$	$\underline{-0.13} \pm 0.20$	-0.49 ± 0.11	1.27 ± 0.53	$\underline{-4.10} \pm 0.03$	-4.12 ± 0.03	-4.29 ± 0.03
Two Moons	$\underline{0.002} \pm 0.001$	0.023 ± 0.002	0.008 ± 0.002	1.40 ± 0.52	$\underline{-0.52} \pm 0.10$	1.36 ± 0.51	-1.02 ± 0.01	$\underline{-0.36} \pm 0.01$	-0.95 ± 0.01
Trajectories	$\underline{0.008} \pm 0.002$	0.016 ± 0.001	0.743 ± 0.005	$\underline{1.03} \pm 0.22$	1.24 ± 0.24	9.30 ± 1.30	$\underline{29.32} \pm 0.02$	26.09 ± 0.02	-215.23 ± 17.6

Table 1: Baseline Comparison – marked in red are cases with notably poor performance; best values are underlined.

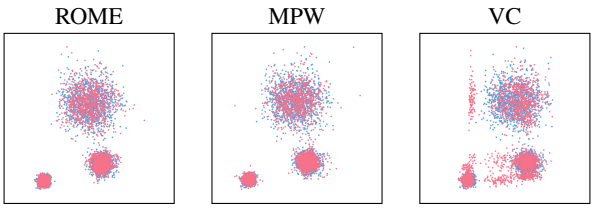

Figure 4: Samples obtained with ROME, MPW and VC (pink) contrasted with samples from p (blue); Varied.

- Three two-dimensional synthetic distributions (Figure 2) were used to test the estimation of distributions with multiple clusters, which might be highly correlated (Aniso) or of varying densities (Varied), or express non-normal distributions (Two Moons).
- A multivariate, 24-dimensional, and highly correlated distribution generated from a subset of the Forking Paths dataset [Liang *et al.*, 2020] (Figure 3). The 24 dimensions correspond to the x and y positions of a human pedestrian across 12 timesteps. Based on 6 original trajectories ($\boldsymbol{x}_i^* \in \mathbb{R}^{12\times 2}$), we defined the underlying distribution p in such a way, that one could calculate a sample $\boldsymbol{x} \sim p$ with:

$$\boldsymbol{x} = s\boldsymbol{x}_i^* \boldsymbol{R}_\theta^T + \boldsymbol{L}\boldsymbol{n}\,, \text{ with } i \sim \mathcal{U}\{1,6\}\,.$$

  Here, $\boldsymbol{R}_\theta \in \mathbb{R}^{2\times 2}$ is a rotation matrix rotating $\boldsymbol{x}_i^*$ by $\theta \sim \mathcal{N}(0, \frac{\pi}{180})$, while $s \sim \mathcal{N}(1, 0.03)$ is a scaling factor. $\boldsymbol{n} = \mathcal{N}(\boldsymbol{0}, 0.03\boldsymbol{I}) \in \mathbb{R}^{12\times 2}$ is additional noise added on all dimensions using $\boldsymbol{L} \in \mathbb{R}^{12\times 12}$, a lower triangular matrix that only contains ones.

Further tests on uni-modal problems can be found in Appendix E.

4.2 Evaluation and Metrics

When estimating density $\widehat{p}$, since we cannot query the distribution p underlying the samples $\boldsymbol{X}$, we require metrics that can provide insights purely based on those samples. To this end we use the following three metrics to quantify how well a given density estimator f can avoid both over-fitting and over-smoothing.

- To test for over-fitting, we first sample two different datasets $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ (N samples each) from p ($\boldsymbol{X}_1, \boldsymbol{X}_2 \sim p$). We then use the estimator f to create two queryable distributions $\widehat{p}_1 = f(\boldsymbol{X}_1)$ and $\widehat{p}_2 = f(\boldsymbol{X}_2)$. If those distributions $\widehat{p}_1$ and $\widehat{p}_2$ are similar, it would mean the tested estimator does not over-fit; we measure this similarity using the Jensen-Shannon divergence [Lin, 1991]:

$$D_{\text{JS}}(\widehat{p}_1 \| \widehat{p}_2) = \frac{1}{2N\ln(2)} \sum_{\boldsymbol{x}\in \boldsymbol{X}_1 \cup \boldsymbol{X}_2} h_1(\boldsymbol{x}) + h_2(\boldsymbol{x})$$

$$h_i(\boldsymbol{x}) = \frac{\widehat{p}_i(\boldsymbol{x})}{\widehat{p}_1(\boldsymbol{x}) + \widehat{p}_2(\boldsymbol{x})} \ln\left(\frac{2\widehat{p}_i(\boldsymbol{x})}{\widehat{p}_1(\boldsymbol{x}) + \widehat{p}_2(\boldsymbol{x})}\right)$$

  This metric, however, is not able to account for systematic biases that could be present in the estimator f. Comparisons of $\widehat{p}_1$ with p – if the dataset allows – can be found in Appendix D.

- To test the goodness-of-fit of the estimated density, we first generate a third set of samples $\widehat{\boldsymbol{X}}_1 \sim \widehat{p}_1$ with N samples. We then use the Wasserstein distance W [Villani, 2009] on the data to calculate the indicator $\widehat{W}$:

$$\widehat{W} = \frac{W(\boldsymbol{X}_1, \widehat{\boldsymbol{X}}_1) - W(\boldsymbol{X}_1, \boldsymbol{X}_2)}{W(\boldsymbol{X}_1, \boldsymbol{X}_2)}$$

  Here, $\widehat{W} > 0$ indicates over-smoothing or misplaced modes, while $-1 \leq \widehat{W} < 0$ indicates over-fitting.

- Not every density estimator f has the ability to generate the samples $\widehat{\boldsymbol{X}}_1$. Consequently, we need to test for goodness-of-fit without relying on $\widehat{\boldsymbol{X}}_1$. Therefore, we use the average log-likelihood

$$\widehat{L} = \frac{1}{N} \sum_{\boldsymbol{x}\in \boldsymbol{X}_2} \ln\left(\widehat{p}_1(\boldsymbol{x})\right)\,,$$

  which would be maximized only for $p = \widehat{p}_1$ as long as $\boldsymbol{X}_2$ is truly representative of p (see Gibbs' inequality [Kvalseth, 1997]). However, using this metric might be meaningless if $\widehat{p}_1$ is not normalized, as the presence of the unknown scaling factor makes the $\widehat{L}$ values of two estimators incomparable, and cannot discriminate between over-fitting and over-smoothing.

For each candidate method f, we used $N = 3000$, and every metric calculation was repeated 100 times to take into account the inherent randomness of sampling from the distributions, with the standard deviation in the tables being reported with $\pm$.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	$-0.13_{\pm 0.20}$	$-0.13_{\pm 0.20}$	$-0.13_{\pm 0.19}$	$-0.14_{\pm 0.20}$
DBCV	$-0.09_{\pm 0.23}$	$-0.09_{\pm 0.23}$	$-0.07_{\pm 0.23}$	$-0.03_{\pm 0.24}$
No clus.	$2.28_{\pm 0.72}$	$2.26_{\pm 0.72}$	$-0.17_{\pm 0.26}$	$10.53_{\pm 2.54}$

Table 2: Ablations ($\widehat{W} \to 0$, Varied): Clustering is essential to prevent over-smoothing. ROME highlighted in gray. Note that the differences between Silhouette and DBCV are not statistically significant.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.
Silhouette	$-2.53_{\pm 0.02}$	$-2.70_{\pm 0.01}$	$-2.79_{\pm 0.01}$
DBCV	$-2.56_{\pm 0.02}$	$-2.69_{\pm 0.01}$	$-2.83_{\pm 0.02}$

Table 3: Ablations ($\widehat{L} \uparrow$, Aniso): When clustering, decorrelation and normalization improve results for distributions with high intra-mode correlation. ROME highlighted in gray.

Cluster.	Norm. Decorr.	Norm. No decorr.	No norm. No decorr.	f_{GMM}
Silhouette	$1.40_{\pm 0.52}$	$1.41_{\pm 0.52}$	$3.65_{\pm 0.99}$	$4.37_{\pm 1.14}$
DBCV	$1.59_{\pm 0.56}$	$1.59_{\pm 0.56}$	$4.24_{\pm 1.13}$	$4.77_{\pm 1.23}$
No clus.	$1.82_{\pm 0.60}$	$1.84_{\pm 0.60}$	$3.17_{\pm 0.89}$	$5.29_{\pm 1.34}$

Table 4: Ablations ($\widehat{W} \to 0$, Two Moons): Excluding normalization or using f_{GMM} as the downstream estimator is not robust against non-normal distributions. ROME highlighted in gray.

4.3 Ablations

To better understand the performance of our approach, we investigated variations in four key aspects of ROME:

- *Clustering approach*. First, we replaced the silhouette score (see Section 3.1) with density based cluster validation (DBCV) [Moulavi *et al.*, 2014] when selecting the optimal clustering out of the 199 possibilities. Furthermore, we investigated the approach of no clustering ($\boldsymbol{C} = \{\{1, \ldots, N\}\}$).
- *Decorrelation vs No decorrelation*. We investigated the effect of removing rotation by setting $\boldsymbol{R}_C = \boldsymbol{I}$.
- *Normalization vs No normalization*. We studied the sensitivity of our approach to normalization by setting $\widetilde{\boldsymbol{\Sigma}}_C = \boldsymbol{I}$.
- *Downstream density estimator*. We replaced f_{KDE} with two other candidate methods. First, we used a single-component Gaussian mixture model f_{GMM}

$$f_{\text{GMM}}(\boldsymbol{X})(\boldsymbol{x}) = \mathcal{N}\left(\boldsymbol{x} | \widehat{\boldsymbol{\mu}}_{\boldsymbol{X}}, \widehat{\boldsymbol{\Sigma}}_{\boldsymbol{X}}\right)$$

fitted to the observed mean $\widehat{\boldsymbol{\mu}}$ and covariance matrix $\widehat{\boldsymbol{\Sigma}}$ of a dataset $\boldsymbol{X}$. Second, we used a k-nearest neighbor approach f_{kNN} [Loftsgaarden and Quesenberry, 1965]

$$f_{\text{kNN}}(\boldsymbol{X})(\boldsymbol{x}) = \frac{k}{N\, \mathcal{V}_M \min_{\widehat{\boldsymbol{x}} \in \boldsymbol{X}} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^M}$$

where $\mathcal{V}_M$ is the volume of the M-dimensional unit hypersphere. We used the rule-of-thumb $k = \lfloor \sqrt{N} \rfloor$. However, this estimator cannot generate samples.

While those four factors would theoretically lead to 24 estimators, f_{KDE} as well as f_{kNN} being invariant against rotation and f_{GMM} being invariant against any linear transformation means that only 14 of ROME's ablations are actually unique.

5 Results

5.1 Baseline Comparison

We found that ROME avoids the major pitfalls displayed by the two baseline methods on the four tested distributions (Table 1). Out of the two baseline methods, the manifold Parzen windows (MPW) approach has a stronger tendency to overfit in the case of the two-dimensional distributions compared to ROME, as quantified by lower D_{JS} values achieved by ROME. MPW does, however, achieve a better log-likelihood for the Two Moons distribution compared to ROME. This could be due to the locally adaptive non-Gaussian distributions being less susceptible to over-smoothing than our approach of using a single isotropic kernel for each cluster if such clusters are highly non-normal. Lastly, in the case of the pedestrian trajectories distribution, ROME once more achieves better performance than MPW, with MPW performing worse both in terms of D_{JS} and $\widehat{L}$.

Meanwhile, the vine copulas (VC) approach tends to over-smooth the estimated densities (large positive $\widehat{W}$ values), and even struggles with capturing the different modes (see Figure 4). This is likely because VC uses KDE with Silverman's rule of thumb, which is known to lead to over-smoothing in the case of multi-modal distributions [Heidenreich *et al.*, 2013]. Furthermore, on the pedestrian trajectories distribution, we observed both high D_{JS} and $\widehat{W}$ values, indicating that VC is unable to estimate the underlying density; this is also indicated by the poor log-likelihood estimates.

Overall, while the baselines were able to achieve better performance in selected cases (e.g., MPW better than ROME in terms of $\widehat{W}$ and $\widehat{L}$ on the Two Moons distribution), they have their apparent weaknesses. Specifically, MPW achieves poor results for most metrics in the case of varying densities within the modes (Varied), while Vine Copulas obtain the worst performance across all three metrics in the case of the multivariate trajectory distributions. ROME, in contrast, achieved high performance across all the test cases.

5.2 Ablation Studies

When it comes to the choice of the clustering method, our experiments show no clear advantage for using either the silhouette score or DBCV. But as the silhouette score is computationally more efficient than DBCV, it is the preferred method.

Clustering	Normalization, Decorrelation f_{KDE}	Normalization, Decorrelation f_{kNN}	Normalization, No decorrelation f_{KDE}	Normalization, No decorrelation f_{kNN}	No normalization f_{KDE}	No normalization f_{kNN}	f_{GMM}
Silhouette	$0.084_{\pm 0.016}$	$1.045_{\pm 0.064}$	$0.777_{\pm 0.116}$	$1.808_{\pm 0.112}$	$0.015_{\pm 0.011}$	$1.887_{\pm 0.118}$	$0.032_{\pm 0.007}$
DBCV	$0.090_{\pm 0.015}$	$1.119_{\pm 0.073}$	$0.897_{\pm 0.154}$	$1.937_{\pm 0.109}$	$0.017_{\pm 0.012}$	$1.934_{\pm 0.116}$	$0.043_{\pm 0.010}$
No clusters	$0.009_{\pm 0.004}$	$0.453_{\pm 0.051}$	$0.015_{\pm 0.012}$	$1.044_{\pm 0.104}$	$0.005_{\pm 0.003}$	$1.478_{\pm 0.132}$	$0.017_{\pm 0.011}$

Table 5: Ablations ($D_{\text{JS}} \downarrow_0^1$, Trajectories; values are multiplied by 10 for easier comprehension): Using f_{kNN} as the downstream estimator tends to lead to over-fitting. ROME highlighted in gray.

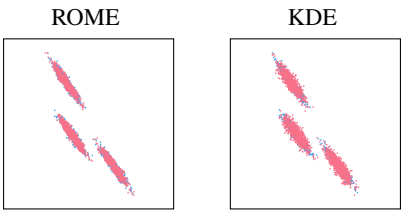

Figure 5: Samples generated by ROME and KDE – equivalent to ROME without clustering, decorrelation and normalization – (in pink) contrasted with samples from p (in blue); Aniso. Note that the samples by KDE are more spread out, indicating over-smoothing.

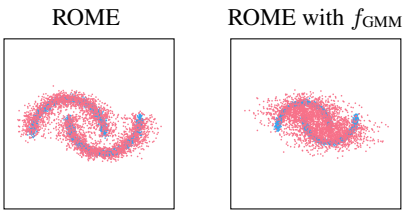

Figure 6: Samples generated by ROME, and ROME with f_{GMM} as the downstream estimator (in pink) contrasted with samples from p (in blue); Two Moons. Note that the samples from f_{GMM} are more spread out, which clearly displays over-smoothing.

However, using clustering is essential, as otherwise there is a risk of over-smoothing, such as in the case of multi-modal distributions with varying densities in each mode (Table 2).

Testing variants of ROME on the Aniso distribution (Table 3) demonstrated not only the need for decorrelation, through the use of rotation, but also normalization in the case of distributions with highly correlated features. There, using either of the two clustering methods in combination with normalization and decorrelation (our full proposed method) is better than the two alternatives of omitting only decorrelation or both decorrelation and normalization. In the case of clustering with the silhouette score, the full method is significantly more likely to reproduce the underlying distribution p by a factor of 1.19 (with a statistical significance of 10^{-50}, see Appendix B) as opposed to omitting only decorrelation, and by 1.30 compared to omitting both decorrelation and normalization. Similar trends can be seen when clustering based on DBCV, with the full method being more likely to reproduce p by a factor of 1.14 and 1.31 respectively. Results on the Aniso distribution further show that KDE on its own is not able to achieve the same results as ROME, but rather it has a tendency to over-smooth (Figure 5). Additionally, the ablation with and without normalization on the Two Moons distribution (Table 4) showed that normalization is necessary to avoid over-smoothing on non-normal distributions.

Lastly, investigating the effect of different downstream density estimators, we found that using ROME with f_{kNN} instead of f_{KDE} leads to over-fitting (highest D_{JS} values in Table 5). Meanwhile, ROME with f_{GMM} tends to over-smooth the estimated density in cases where the underlying distribution is not Gaussian (high $\widehat{W}$ in Table 4). The over-smoothing caused by f_{GMM} is further visualised in Figure 6.

In conclusion, our ablation studies confirmed that using f_{KDE} in combination with data clustering, normalization and decorrelation provides the most reliable density estimation for different types of distributions.

6 Conclusion

In our comparison against two established and sophisticated density estimators, we observed that ROME achieved consistently good results across all tests, while manifold Parzen windows (MPW) and vine copulas (VC) were susceptible to over-fitting and over-smoothing. For example, while MPW is superior at capturing non-normal distributions, it produces kernels with too small of a bandwidth (hence the over-fitting), which is likely caused by the selected number of nearest neighbors used for the localized kernel estimation being too small. Meanwhile, compared to VC, ROME is numerically more stable and does not hallucinate new modes in the estimated densities (Figure 4). Furthermore, as part of several ablation studies, we found that ROME overcomes the shortcomings of other common density estimators, such as the over-fitting exhibited by kNN or the over-smoothing by GMM. In those studies, we additionally demonstrated that our approach of using clustering, decorrelation, and normalization is indispensable for overcoming the deficiencies of KDE.

Future work can further improve on our results by investigating the integration of more sophisticated density estimation methods, such as MPW, instead of the kernel density estimator in our proposed approach to enable better performance on non-normal clusters.

Overall, by providing a simple way to accurately estimate distributions based on samples, ROME can help in better handling and evaluating probabilistic data as well as enabling more precise probabilistic inference.

Acknowledgments

This research was supported by NWO-NWA project "Acting under uncertainty" (ACT), NWA.1292.19.298.

Contribution Statement

Anna Mészáros and Julian F. Schumann equally contributed to the design and implementation of the research, to the analysis of the results and to the writing of the manuscript and should be considered joint first authors. Javier Alonso-Mora provided valuable feedback on the writing of the manuscript. Arkady Zgonnikov and Jens Kober provided valuable feedback at all steps of the project and should be considered joint last authors.

References

[Ankerst *et al.*, 1999] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *ACM Sigmod Record*, 28(2):49–60, 1999.

[Bakam and Pommeret, 2023] Yves I Ngounou Bakam and Denys Pommeret. Nonparametric estimation of copulas and copula densities by orthogonal projections. *Econometrics and Statistics*, 2023.

[Deisenroth *et al.*, 2020] Marc Peter Deisenroth, A Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[Ester *et al.*, 1996] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231, 1996.

[Gao *et al.*, 2022] Jia-Xing Gao, Da-Quan Jiang, and Min-Ping Qian. Adaptive manifold density estimation. *Journal of Statistical Computation and Simulation*, 92(11):2317–2331, 2022.

[Goodfellow *et al.*, 2020] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[Heidenreich *et al.*, 2013] Nils-Bastian Heidenreich, Anja Schindler, and Stefan Sperlich. Bandwidth selection for kernel density estimation: a review of fully automatic selectors. *AStA Advances in Statistical Analysis*, 97:403–433, 2013.

[Joe, 2014] Harry Joe. *Dependence modeling with copulas*. CRC press, 2014.

[Kobyzev *et al.*, 2020] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, 2020.

[Kvalseth, 1997] Tarald O Kvalseth. Generalized divergence and gibbs' inequality. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, volume 2, pages 1797–1801. IEEE, 1997.

[Liang *et al.*, 2020] Junwei Liang, Lu Jiang, Kevin Murphy, Ting Yu, and Alexander Hauptmann. The garden of forking paths: Towards multi-future trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10508–10518, 2020.

[Lin, 1991] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[Loftsgaarden and Quesenberry, 1965] Don O Loftsgaarden and Charles P Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, 1965.

[McLachlan and Rathnayake, 2014] Geoffrey J McLachlan and Suren Rathnayake. On the number of components in a gaussian mixture model. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(5):341–355, 2014.

[Moulavi *et al.*, 2014] Davoud Moulavi, Pablo A Jaskowiak, Ricardo JGB Campello, Arthur Zimek, and Jörg Sander. Density-based clustering validation. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 839–847. SIAM, 2014.

[Mozaffari *et al.*, 2020] Sajjad Mozaffari, Omar Y Al-Jarrah, Mehrdad Dianati, Paul Jennings, and Alexandros Mouzakitis. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):33–47, 2020.

[Nagler and Czado, 2016] Thomas Nagler and Claudia Czado. Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *Journal of Multivariate Analysis*, 151:69–89, 2016.

[Otneim and Tjøstheim, 2017] Håkon Otneim and Dag Tjøstheim. The locally gaussian density estimator for multivariate data. *Statistics and Computing*, 27:1595–1616, 2017.

[Parzen, 1962] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[Rasouli, 2020] Amir Rasouli. Deep learning for vision-based prediction: A survey. *arXiv preprint arXiv:2007.00095*, 2020.

[Rousseeuw, 1987] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[Silverman, 1998] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 1998.

[Villani, 2009] Cédric Villani. *Optimal transport: Old and new*, volume 338. Springer, 2009.

[Vincent and Bengio, 2002] Pascal Vincent and Yoshua Bengio. Manifold parzen windows. *Advances in Neural Information Processing Systems*, 15, 2002.

[Wang *et al.*, 2009] Xiaoxia Wang, Peter Tino, Mark A Fardal, Somak Raychaudhury, and Arif Babul. Fast parzen window density estimator. In *2009 International Joint Conference on Neural Networks*, pages 3267–3274. IEEE, 2009.

[Wei *et al.*, 2020] Ruoqi Wei, Cesar Garcia, Ahmed El-Sayed, Viyaleta Peterson, and Ausif Mahmood. Variations in variational autoencoders-a comparative evaluation. *IEEE Access*, 8:153651–153670, 2020.

[Wold *et al.*, 1987] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[Xu *et al.*, 2019] Donna Xu, Yaxin Shi, Ivor W Tsang, Yew-Soon Ong, Chen Gong, and Xiaobo Shen. Survey on multi-output learning. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7):2409–2429, 2019.