Journal of Machine Learning Research 26 (2025) 1-64 Submitted 6/24; Revised 3/25; Published 6/25

Learning conditional distributions on continuous spaces

Cyril Bénézet[*†] CYRIL.BENEZET@ENSIIE.FR
Université Paris-Saclay, CNRS, Univ Evry, ensIIE
Laboratoire de Mathématiques et Modélisation d'Evry,
91037, Evry-Courcouronnes, France

Ziteng Cheng[*‡] ZITENG.CHENG@UTORONTO.CA
Financial Technology Thrust
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, 511400, China

Sebastian Jaimungal[*§] SEBASTIAN.JAIMUNGAL@UTORONTO.CA
Department of Statistical Sciences
University of Toronto
Toronto, ON M5G 1Z5, Canada

Editor: Maxim Raginsky

Abstract

We investigate sample-based learning of conditional distributions on multi-dimensional unit boxes, allowing for different dimensions of the feature and target spaces. Our approach involves clustering data near varying query points in the feature space to create empirical measures in the target space. We employ two distinct clustering schemes: one based on a fixed-radius ball and the other on nearest neighbors. We establish upper bounds for the convergence rates of both methods and, from these bounds, deduce optimal configurations for the radius and the number of neighbors. We propose to incorporate the nearest neighbors method into neural network training, as our empirical analysis indicates it has better performance in practice. For efficiency, our training process utilizes approximate nearest neighbors search with random binary space partitioning. Additionally, we employ the Sinkhorn algorithm and a sparsity-enforced transport plan. Our empirical findings demonstrate that, with a suitably designed structure, the neural network has the ability to adapt to a suitable level of Lipschitz continuity locally. For reproducibility, our code is available at `https://github.com/zcheng-a/LCD_kNN`.

Keywords: non-parametric statistics, Wasserstein distance, deep learning, Lipschitz continuity

1. Introduction

Learning the conditional distribution is a crucial aspect of many decision-making scenarios. While this learning task is generally challenging, it presents unique complexities when explored in a continuous space setting. Below, we present a classic example (cf. Booth et al. (1992); Pflug and Pichler (2016)) that highlights this core challenge.

∗. Ordered alphabetically.

†. CB gratefully acknowledges financial support from the Agence Nationale de la Recherche (ReLISCoP grant ANR-21-CE40- 0001).

‡. The majority of ZC's work was completed during the postdoctoral appointment in the Department of Statistical Sciences at the University of Toronto. ZC gratefully acknowledges financial support from the Guangzhou-HKUST(GZ) Joint Funding Program (Grant No. 2024A03J0630).

§. SJ would like acknowledge support from the Natural Sciences and Engineering Research Council of Canada (grants RGPIN-2024-04317, ALLRP 550308-20, and RGPIN-2020-04289).

For simplicity, we suppose the following model

$$Y = \tfrac{1}{2}X + \tfrac{1}{2}U,$$

where the feature variable X and the noise U are independent Uniform($[0,1]$), and Y is the target variable. Upon collecting a finite number of independent samples $\mathcal{D} = \{(X_m, Y_m)\}_{m=1}^M$, we aim to estimate the conditional distribution of Y given X. Throughout, we treat this conditional distribution as a measure-valued function of x, denoted by P_x. A naive approach is to first form an empirical joint measure

$$\hat{\psi} := \frac{1}{M}\sum_{m=1}^{M} \delta_{(X_m, Y_m)},$$

where δ stands for the Dirac meaasure, and then use the conditional distribution induced from $\hat{\psi}$ as an estimator. As the marginal distribution of X is continuous, with probability 1 (as $\mathbb{P}(X_m = X_{m'}) = 0$ for all $m \neq m'$), we have that[1]

$$\widehat{P}_x = \begin{cases} \delta_{Y_m}, & x = X_m \text{ for some } m, \\ \text{Uniform}([0,1]), & \text{otherwise.} \end{cases}$$

Regardless of the sample size M, $\widehat{P}_x$ fails to approximate the true conditional distribution,

$$P_x = \text{Uniform}\left([x, x+\tfrac{1}{2}]\right), \quad x \in [0,1].$$

Despite the well-known convergence of the (joint) empirical measure to the true distribution Dudley (1969); Fournier and Guillin (2015), the resulting conditional distribution often fails to provide an accurate approximation of the true distribution. This discrepancy could be due to the fact that calculating conditional distribution is an inherently unbounded operation. As a remedy, clustering is a widely employed technique. Specifically, given a query point x in the feature space, we identify samples where X_m is close to x and use the corresponding Y_m's to estimate P_x. Two prominent methods within the clustering approach are the kernel method and the nearest neighbors method[2]. Roughly speaking, the kernel method relies primarily on proximity to the query point for selecting X_m's, while the nearest neighbors method focuses on the rank of proximity. Notably, discretizing the feature space (also known as quantization), a straightforward yet often effective strategy, can be seen as a variant of the kernel method with static query points and flat kernels.

The problem of estimating conditional distributions can be addressed within the non-parametric regression framework, by employing clustering or resorting to non-parametric least squares, among others. Alternatively, it is feasible to estimate the conditional density function directly: a widely-used method involves estimating the joint and marginal density functions using kernel smoothing and then calculating their ratio. This method shares similarities with the clustering heuristics mentioned earlier. For a more detailed review of these approaches, we refer to Section 1.2.

This work draws inspiration from recent advancements in estimating discrete-time stochastic processes using conditional density function estimation (Pflug and Pichler (2016)) and quantization methods (Backhoff et al. (2022); Acciaio and Hou (2023)). A notable feature of these works is their use of the Wasserstein distance to calculate local errors: the difference between the true and estimated conditional distributions at a query point x. One could average these local errors across

1. In accordance to the model, we set the conditional distribution to Uniform($[0,1]$) at points where it is not well-defined.
2. These should not be confused with similarly named methods used in density function estimation.

different values of x's to gauge the global error. Employing Wasserstein distances naturally frames the study within the context of weak convergence, thereby enabling discussions in a relatively general setting, although this approach may yield somewhat weaker results in terms of the mode of convergence. Moreover, utilizing a specific distance rather than the general notion of weak convergence enables a more tangible analysis of the convergence rates and fluctuations. We would like to point out that the advancements made in Backhoff et al. (2022); Acciaio and Hou (2023), as well as our analysis in this paper, relies on recent developments concerning the Wasserstein convergence rate of empirical measures under i.i.d. sampling from a static distribution (cf. Fournier and Guillin (2015)).

1.1 Main contributions

First, we introduce some notations to better illustrate the estimators that we study. Let $\mathbb{X}$ and $\mathbb{Y}$ be multi-dimensional unit cubes, with potentially different dimensions, for feature and target spaces. For any integer $M \geq 1$, any $\mathsf{D} = \{(x_m, y_m)\}_{m=1}^M \in (\mathbb{X} \times \mathbb{Y})^M$, and any Borel set $A \subset \mathbb{X}$, we define a probability measure on $\mathbb{Y}$ by

$$\hat{\mu}_A^{\mathsf{D}} := \begin{cases} \left(\sum_{m=1}^M \mathbb{1}_A(x_m)\right)^{-1} \sum_{m=1}^M \mathbb{1}_A(x_m)\delta_{y_m}, & \sum_{m=1}^M \mathbb{1}_A(x_m) > 0, \\ \lambda_{\mathbb{Y}}, & \text{otherwise}, \end{cases} \tag{1}$$

where $\lambda_{\mathbb{Y}}$ is the Lebesgue measure on $\mathbb{Y}$ and, for $y \in \mathbb{Y}$, δ_y is a Dirac measure with atom at y. In general, one could consider weighting δ_{y_m}'s (cf. (Györfi et al., 2002, Section 5), (Biau and Devroye, 2015, Chapter 5)), which may offer additional benefits in specific applications. As such adjustments are unlikely to affect the convergence rate, however, we use uniform weighting for simplicity.

With (random) data $\mathcal{D} = \{(X_m, Y_m)\}_{m=1}^M$, we aim to estimate the conditional distribution of Y given X. We view this conditional distribution as a measure-valued function $P : \mathbb{X} \to \mathcal{P}(\mathbb{Y})$ and use a subscript for the input argument and write P_x. Consider a clustering scheme[3] given by the map $\mathcal{A}^{\mathsf{D}} : \mathbb{X} \to 2^{\mathbb{X}}$. We investigate estimators of the form $x \mapsto \hat{\mu}^{\mathcal{D}}_{\mathcal{A}^{\mathcal{D}}(x)}$. We use $\widehat{P}^{\mathcal{A}}$ to denote said estimator and suppress $\mathcal{D}$ from the notation for convenience. In later sections, we consider two kinds of maps $\mathcal{A}^{\mathsf{D}}$ (i) a ball with fixed radius centered at x, called an r-box and (ii) the k nearest neighbors of x, called k-nearest-neighbor estimator. See Definitions 5 and 9 for more details.

One of our main contribution pertains to analyzing the error

$$\int_{\mathbb{X}} \mathcal{W}\left(P_x, \widehat{P}_x^{\mathcal{A}}\right) \, \nu(\mathrm{d}x), \tag{2}$$

where $\mathcal{W}$ is the 1-Wasserstein distance (see (Villani, 2008, Particular Case 6.2)) and $\nu \in \mathcal{P}(\mathbb{X})$ is arbitrary and provides versatility to the evaluation criterion. A canonical choice for ν is the Lebesgue measure on $\mathbb{X}$, denoted by $\lambda_{\mathbb{X}}$. This is particularly relevant in control settings where $\mathbb{X}$ represents the state-action space and accurate approximations across various state and action scenarios are crucial for making informed decisions. The form of error above is also foundational in stochastic process estimation under the adapted Wasserstein distance (cf. (Backhoff et al., 2022, Lemma 3.1)), making the techniques we develop potentially relevant in other contexts. Under the assumption that P is Lipschitz continuous (Assumption 2) and standard assumptions on the data collection process (Assumption 3), we analyze the convergence rate and fluctuation by bounding

3. In general, the clustering scheme may require information on $(x_1, \ldots, x_M)$. For example, clustering the k-nearest-neighbor near a query point x requires to know all x_m's.

the following two quantities

$$\mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \widehat{P}_x^{\mathcal{A}}\right) \nu(\mathrm{d}x)\right] \quad \text{and} \quad \mathrm{Var}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \widehat{P}_x^{\mathcal{A}}\right) \nu(\mathrm{d}x)\right].$$

Moreover, by analyzing the above quantities, we gain insights into the optimal choice of the clustering mapping $\mathcal{A}$. For the detail statements of these results, we refer to Theorems 7, 10, 8, and 11. We also refer to Section 2.4 for related comments.

To illustrate another aspect of our contribution, we note by design $x \mapsto \widehat{P}_x^{\mathcal{A}}$ is piece-wise constant. This characteristic introduces limitations. Notably, it renders the analysis of performance at the worst-case x elusive. Contrastingly, by building a Lipschitz-continuous parametric estimator $\tilde{P}^{\Theta}$ from the raw estimator $\widehat{P}^{\mathcal{A}}$, in Proposition 13 we demonstrate that an upper bound on the aforementioned expectation allows us to derive a worst-case performance guarantee. Guided by Proposition 13, we explore a novel approach of training a neural network for estimation, by using $\widehat{P}^{\mathcal{A}}$ as training data and incorporating suitably imposed Lipschitz continuity. To be comprehensive, we include in Section 1.2 a review of studies on Lipschitz continuity in neural networks.

In Section 3.1, we define $\tilde{P}^{\theta}$ as a neural network that approximates P, where θ represents the network parameters. We train $\tilde{P}^{\theta}$ with the objective:

$$\arg\min_{\theta} \sum_{n=1}^{N} \mathcal{W}\left(\widehat{P}^{\mathcal{A}}_{\tilde{X}_n}, \tilde{P}^{\theta}_{\tilde{X}_n}\right),$$

where $(\tilde{X}_n)_{n=1}^N$ is a set of randomly selected query points. For implementation purposes, we use the k-nearest-neighbor estimator in the place of $\widehat{P}^{\mathcal{A}}$ (see Definition 9). To mitigate the computational costs stemming from the nearest neighbors search, we employ the technique of Approximate Nearest Neighbor Search with Random Binary Space Partitioning (ANN-RBSP), as discussed in Section 3.1.1. In Section 3.1.2, we compute $\mathcal{W}$ using the Sinkhorn algorithm, incorporating normalization and enforcing sparsity for improved accuracy. To impose a suitable level of local Lipschitz continuity on $\tilde{P}^{\theta}$, in Section 3.1.3, we employ a neural network with a specific architecture and train the networks using a tailored procedure. The key component of this architecture is the convex potential layer introduced in Meunier et al. (2022). In contrast to most extant literature that imposes Lipschitz continuity on neural networks, our approach does not utilize specific constraint or regularization of the objective function, but relies on certain self-adjusting mechanism embedded in the training.

In Section 3.2, we evaluate the performance of the trained $\tilde{P}^{\theta}$, denoted by $\tilde{P}^{\Theta}$, using three sets of synthetic data in 1D and 3D spaces. Our findings indicate that $\tilde{P}^{\Theta}$ generally outperforms $\widehat{P}^{\mathcal{A}}$, even though it is initially trained to match $\widehat{P}^{\mathcal{A}}$. This superior performance persists even when comparing $\tilde{P}^{\Theta}$ to different $\widehat{P}^{\mathcal{A}}$ using various clustering parameters, without retraining $\tilde{P}^{\Theta}$. Furthermore, despite using the same training parameters, $\tilde{P}^{\Theta}$ consistently demonstrates the ability to adapt to a satisfactory level of local Lipschitz continuity across all cases. Moreover, in one of the test cases, we consider a kernel that exhibits a jump discontinuity, and we find that $\tilde{P}^{\Theta}$ handles this jump case well despite Lipschitz continuity does not hold.

Lastly, we provide further motivation of our approach by highlighting some potential applications for $\tilde{P}^{\Theta}$. The first application is in model-based policy gradient method in reinforcement learning. We anticipate that the enforced Lipschitz continuity allows us to directly apply the policy gradient update via compositions of $\tilde{P}^{\Theta}$ and cost function for more effective optimality searching. The second application of $\tilde{P}^{\Theta}$ is in addressing optimisation in risk-averse Markov decision processes, where dynamic programming requires knowledge beyond the conditional expectation of the risk-to-go (cf. Chow et al. (2015); Huang and Haskell (2017); Coache et al. (2023); Cheng and Jaimungal (2023)). The study of these applications is left for further research.

1.2 Related works

In this section, we will first review the clustering approach in estimating conditional distributions, and then proceed to review recent studies on Lipschitz continuity in neural networks.

1.2.1 Estimating conditional distributions via clustering

The problem of estimating conditional distributions is frequently framed as non-parametric regression problems for real-valued functions. For instance, when $d_{\mathbb{Y}} = 1$, estimate the conditional α-quantile of Y given X. Therefore, we begin by reviewing some of the works in non-parametric regression.

The kernel method in non-parametric regression traces its origins back to the Nadaraya-Watson estimator (Nadaraya (1964); Watson (1964)), if not earlier. Subsequent improvements have been introduced, such as integral smoothing (Gasser and Müller (1979), also known as the Gasser-Müller estimator), local fitting with polynomials instead of constants (Fan (1992)), and adaptive kernels (Hall et al. (1999)). Another significant area of discussion is the choice of kernel bandwidth, as detailed in works like Hardle and Marron (1985); Gijbels and Goderniaux (2004); Köhler et al. (2014). Regarding convergence rates, analyses under various settings can be found in Stone (1982); Hall and Hart (1990); Kohler et al. (2009, 2010), with Stone (1982) being particularly relevant to our study for comparative purposes. According to Stone (1982), if the target function is Lipschitz continuous, with i.i.d. sampling and that the sampling distribution in the feature space has a uniformly positive density, then the optimal rate of the $\|\cdot\|_1$-distance between the regression function and the estimator is of the order $M^{-\frac{1}{d_{\mathbb{X}}+2}}$. For a more comprehensive review of non-parametric regression using kernel methods, we refer to the books such as Györfi et al. (2002); Ferraty and Vieu (2006); Wasserman (2006) and references therein.

Non-parametric regression using nearest neighbors methods originated from classification problems (Fix and Hodges (1951)). Early developments in this field can be found in Mack (1981); Devroye (1982); Bhattacharya and Gangopadhyay (1990). For a comprehensive introduction to nearest neighbors methods, we refer to Györfi et al. (2002). More recent reference such as Biau and Devroye (2015) offers further detailed exploration of the topic. The nearest neighbor method can be viewed as a variant of the kernel method that adjusts the bandwidth based on the number of local data points—a property that has gained significant traction. Recently, the application of the nearest neighbor method has expanded into various less standard settings, including handling missing data (Rachdi et al. (2021)), reinforcement learning (Shah and Xie (2010); Giegrich et al. (2024)), and time series forecasting (Martínez et al. (2017)). For recent advancements in convergence analysis beyond the classical setting, see Zhao and Lai (2019); Padilla et al. (2020); Ryu and Kim (2022); Demirkayaa et al. (2024).

Although the review above mostly focuses on clustering approach, other effective approaches exist, such as non-parametric least square, or more broadly, conditional elicitability (e.g., Györfi et al. (2002); Muandet et al. (2017); Wainwright (2019); Coache et al. (2023)). Non-parametric least square directly fits the data using a restricted class of functions. At first glance, this approach appears distinct from clustering. However, they share some similarities in their heuristics: the rigidity of the fitting function, due to imposed restrictions, allows data points near the query point to affect the estimation, thereby implicitly incorporating elements of clustering.

Apart from non-parametric regression, conditional density function estimation is another significant method for estimating conditional distributions. One approach is based on estimating joint and marginal density functions, and then using the ratio of these two to produce an estimator for the conditional density function. A key technique used in this approach is kernel smoothing. Employing a static kernel for smoothing results in a conditional density estimator that shares similar clustering

heuristics to those found in the kernel method of non-parametric regression. For a comprehensive overview of conditional density estimation, we refer to reference books such as Scott (2015); Simonoff (1996). For completeness, we also refer to (Muandet et al., 2017, Section 5.1) for a perspective on static density function estimation from the standpoint of reproducing kernel Hilbert space. Further discussions on estimation using adaptive kernels can be found in, for example, Bashtannyk and Hyndman (2001); Lacour (2007); Bertin et al. (2016); Zhao and Tabak (2023).

Despite extensive research in non-parametric regression and conditional density function estimation, investigations from the perspective of weak convergence have been relatively limited, only gaining more traction in the past decade. Below, we highlight a few recent studies conducted in the context of estimating discrete-time stochastic processes under adapted Wasserstein distance, as the essence of these studies are relevant to our evaluation criterion (2). Pflug and Pichler (2016) explores the problem asymptotically, employing tools from conditional density function estimation with kernel smoothing. Subsequently, Backhoff et al. (2022) investigates a similar problem with a hypercube as state space, employing the quantization method. Their approach removes the need to work with density functions. They calculate the convergence rate, by leveraging recent developments in the Wasserstein convergence rate of empirical measures Fournier and Guillin (2015). Moreover, a sub-Gaussian concentration with parameter M^{-1} is established. The aforementioned results are later extended to $\mathbb{R}^d$ in Acciaio and Hou (2023), where a non-uniform grid is used to mitigate assumptions on moment conditions. Most recently, Hou (2024) examines smoothed variations of the estimators proposed in Backhoff et al. (2022); Acciaio and Hou (2023). Other developments on estimators constructed from smoothed quantization can be found in Šmíd and Kozmík (2024).

Lastly, regarding the machine learning techniques used in estimating conditional distributions, conditional generative models are particularly relevant. For reference, see Mirza and Osindero (2014); Papamakarios et al. (2017); Vaswani et al. (2017); Fetaya et al. (2020). These models have achieved numerous successes in image generation and natural language processing. We suspect that, due to the relatively discrete (albeit massive) feature spaces in these applications, clustering is implicitly integrated into the training procedure. In continuous spaces, under suitable setting, clustering may also become an embedded part of the training procedure. For example, implementations in Li et al. (2020b); Vuletić et al. (2024); Hosseini et al. (2024) do not explicitly involve clustering and use training objectives that do not specifically address the issues highlighted in the motivating example at the beginning of the introduction. Their effectiveness could possibly be attributed to certain regularization embedded within the neural network and training procedures. Nevertheless, research done in continuous spaces that explicitly uses clustering approaches when training conditional generative models holds merit. Such works are relatively scarce. For an example of this limited body of research, we refer to Xu and Acciaio (2022), where the conditional density function estimator from Pflug and Pichler (2016) is used to train an adversarial generative network for stochastic process generation.

1.2.2 Lipschitz continuity in neural networks

Recently, there has been increasing interest in understanding and enforcing Lipschitz continuity in neural networks. The primary motivation is to provide a certifiable guarantee for classification tasks performed by neural networks: it is crucial that minor perturbations in the input object have a limited impact on the classification outcome.

One strategy involves bounding the Lipschitz constant of a neural network, which can then be incorporated into the training process. For refined upper bounds on the (global) Lipschitz constant, see, for example, Bartlett et al. (2017); Virmaux and Scaman (2018); Tsuzuku et al. (2018); Fazlyab et al. (2019); Xue et al. (2022); Fazlyab et al. (2024). For local bounds, we refer to Jordan and

Dimakis (2021); Bhowmick et al. (2021); Shi et al. (2022) and the references therein. We also refer to Zhang et al. (2022) for a study of the Lipschitz property from the viewpoint of boolean functions.

Alternatively, designing neural network architectures that inherently ensure desirable Lipschitz constants is another viable strategy. Works in this direction include Meunier et al. (2022); Singla et al. (2022); Wang and Manchester (2023); Araujo et al. (2023). Notably, the layer introduced in Meunier et al. (2022) belongs to the category of residual connection (He et al. (2016)).

Below, we review several approaches that enforce Lipschitz constants during neural network training. Tsuzuku et al. (2018); Liu et al. (2022) explore training with a regularized objective function that includes upper bounds on the network's Lipschitz constant. Gouk et al. (2021) frame the training problem into constrained optimization and train with projected gradients descent. Given the specific structure of the refined bound established in Fazlyab et al. (2019), Pauli et al. (2022) combines training with semi-definite programming. They develop a version with a regularized objective function and another that enforces the Lipschitz constant exactly. Fazlyab et al. (2024) also investigates training with a regularized objective but considers Lipschitz constants along certain directions. Huang et al. (2021) devises a training procedure that removes components from the weight matrices to achieve smaller local Lipschitz constants. Trockman and Kolter (2021) initially imposes orthogonality on the weight matrices, and subsequently enforces a desirable Lipschitz constant based on that orthogonality. Ensuring desirable Lipschitz constants with tailored architectures, Singla et al. (2022); Wang and Manchester (2023) train the networks directly. Although the architecture proposed in Meunier et al. (2022) theoretically ensures the Lipschitz constant, it requires knowledge of the spectral norm of the weight matrices, which does not admit explicit expression in general. Their training approach combines power iteration for spectral norm approximation with the regularization methods used in Tsuzuku et al. (2018).

Finally, we note that due to their specific application scenarios, these implementations concern relatively stringent robustness requirements and thus necessitate more specific regularization or constraints. In our setting, it is generally desirable for the neural network to automatically adapt to a suitable level of Lipschitz continuity based on the data, while also avoiding excessive oscillations from over-fitting. The literature directly addressing this perspective is limited (especially in the setting of conditional distribution estimation). We refer to Bai et al. (2021); Bountakas et al. (2023); Cohen et al. (2019) for discussions that could be relevant.

1.3 Organization of the paper

Our main theoretical results are presented in Section 2. Section 3 is dedicated to the training of $\tilde{P}^{\Theta}$. We will outline the key components of our training algorithm and demonstrate its performance on three sets of synthetic data. We will prove the theoretical results in Section 4. Further implementation details and ablation analysis are provided in Section 5. In Section 6, we discuss the weaknesses and potential improvements of our implementation. Appendix B and C respectively contain additional plots and a table that summarizes the configuration of our implementation. Additionally, Appendix D includes a rougher version of the fluctuation results.

Notations and terminologies

Throughout, we adopt the following set of notations and terminologies.

- On any normed space $(E, \|\cdot\|)$, for all $x \in E$ and $\gamma > 0$, $B(x, \gamma)$ denotes the closed ball of radius γ around x, namely $B(x, \gamma) = \{x' \in E \mid \|x - x'\| \leq \gamma\}$.
- For any measurable space $(E, \mathscr{E})$, $\mathcal{P}(E)$ denotes the set of probability distributions on $(E, \mathscr{E})$. For all $x \in E$, $\delta_x \in \mathcal{P}(E)$ denotes the Dirac mass at x.

- We endow normed spaces $(E, \|\cdot\|)$ with their Borel sigma-algebra $\mathscr{B}(E)$, and $\mathcal{W}$ denotes the 1-Wasserstein distance on $\mathcal{P}(E)$.

- On $\mathbb{X} = [0,1]^d$, we denote by $\lambda_{\mathbb{X}}$ the Lebesgue measure. We say a measure $\nu \in \mathcal{P}(\mathbb{X})$ is dominated by Lebesgue measure with a constant $\overline{C} > 0$ if $\nu(A) \leq \overline{C}\lambda_{\mathbb{X}}(A)$ for all $A \in \mathscr{B}([0,1]^d)$.

- The symbol $\sim$ denotes equivalence in the sense of big O notation, indicating that each side dominates the other up to a multiplication of some positive absolute constant. More precisely, $a_n \sim b_n$ means there are finite constants $c, C > 0$ such that

$$c\, a_n \leq b_n \leq C\, a_n, \quad n \in \mathbb{N}.$$

  Similarly, $\lesssim$ implies that one side is of a lesser or equal, in the sense of big O notation, compared to the other.

2. Theoretical results

In Section 2.1, we first formally set up the problem and introduce some technical assumption. We then study in Section 2.2 and 2.3 the convergence and fluctuation of two versions of $\widehat{P}^{\mathcal{A}}$, namely, the r-box estimator and the k-nearest-neighbor estimator. Related comments are organized in Section 2.4. Moreover, in Section 2.5, we provide a theoretical motivation for the use of $\tilde{P}^{\Theta}$, the Lipschitz-continuous parametric estimator trained from $\widehat{P}^{\mathcal{A}}$.

2.1 Setup

For $d_{\mathbb{X}}, d_{\mathbb{Y}} \geq 1$ two integers, we consider $\mathbb{X} := [0,1]^{d_{\mathbb{X}}}$ and $\mathbb{Y} := [0,1]^{d_{\mathbb{Y}}}$, endowed with their respective sup-norm $\|\cdot\|_\infty$.

Remark 1 *The sup-norm is chosen for simplicity of the theoretical analysis only: as all norms on $\mathbb{R}^n$ are equivalent (for any generic $n \geq 1$), our results are valid, up to different multiplicative constants, for any other choice of norm.*

We aim to estimate an unknown probabilistic kernel

$$\begin{aligned} P : \mathbb{X} &\to \mathcal{P}(\mathbb{Y}) \\ x &\mapsto P_x(\mathrm{d}y). \end{aligned}$$

To this end, given an integer-valued sample size $M \geq 1$, we consider a set of (random) data points $\mathcal{D} := \{(X_m, Y_m)\}_{m=1}^M$ associated to P. We also define the set of projections of the data points onto the feature space as $\mathcal{D}_{\mathbb{X}} := \{X_m\}_{m=1}^M$.

Throughout this section, we work under the following technical assumptions.

Assumption 2 (Lipschitz continuity of kernel) *There exists $L \geq 0$ such that, for all $(x, x') \in \mathbb{X}^2$,*

$$\mathcal{W}(P_x, P_{x'}) \leq L\|x - x'\|_\infty.$$

Assumption 3 *The following is true:*

(i) $\mathcal{D}$ *is i.i.d. with probability distribution* $\psi := \xi \otimes P$*, where* $\xi \in \mathcal{P}(\mathbb{X})$ *and where* $\xi \otimes P \in \mathcal{P}(\mathbb{X} \times \mathbb{Y})$ *is (uniquely, by Caratheodory extension theorem) defined by*

$$(\xi \otimes P)(A \times B) := \int_{\mathbb{X}} \mathbb{1}_A(x) P_x(B) \xi(\mathrm{d}x), \quad A \in \mathscr{B}(\mathbb{X}), B \in \mathscr{B}(\mathbb{Y}).$$

(ii) There exists $\underline{c} \in (0, 1]$ *such that, for all* $A \in \mathscr{B}(\mathbb{X})$*,* $\xi(A) \geq \underline{c}\, \lambda_{\mathbb{X}}(A)$*.*

These assumptions allow us to analyze convergence and gain insights into the optimal clustering hyper-parameters without delving into excessive technical details. Assumption 2 is mainly used for determining the convergence rate. If the convergence rate is not of concern, it is possible to establish asymptotic results with less assumptions. We refer to Devroye (1982); Backhoff et al. (2022) for relevant results. The conditions placed on ξ in Assumption 3 are fairly standard, though less stringent alternatives are available. For instance, Assumption 3 (i) can be weakened by considering suitable dependence Hall and Hart (1990) or ergodicity in the context of stochastic processes Rudolf and Schweizer (2018). Assumption 3 (ii), implies there is mass almost everywhere and is aligned with the motivation from control settings discussed in the introduction. Assumptions 2 and 3 are not exceedingly stringent and provides a number of insights into the estimation problem. More general settings are left for further research.

The estimators discussed in subsequent sections are of the form $\widehat{P}^{\mathcal{A}}$, as introduced right after (1), for two specific choices of clustering schemes $\mathcal{A}$ constructed with the data $\mathcal{D}$.

Remark 4 *In the following study, we assert all the measurability needed for* $\widehat{P}^{\mathcal{A}}$ *to be well-defined. These measurability can be verified using standard measure-theoretic tools listed in, for example, (Aliprantis and Border, 2006, Section 4 and 15).*

2.2 Results on r-box estimator

The first estimator, which we term the r-box estimator, is defined as follows.

Definition 5 *Choose* r*, a real number, s.t.* $0 < r < \frac{1}{2}$*. The* r*-box estimator for* P *is defined by*

$$\begin{aligned} \hat{P}^r : \mathbb{X} &\to \mathcal{P}(\mathbb{Y}) \\ x &\mapsto \hat{P}^r_x := \hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \end{aligned}$$

where, for all $x \in \mathbb{X}$*,* $\mathcal{B}^r(x) := B(\beta^r(x), r)$ *and* $\beta^r(x) := r \vee x \wedge (1 - r)$*, where* $r \vee \cdot$ *and* $\cdot \wedge (1 - r)$ *are applied entry-wise.*

Remark 6 *The set* $\mathcal{B}^r(x)$ *is defined such that it is a ball of radius around* x *whenever* x *is at least* r *away from the boundary* $\partial\mathbb{X}$ *(in all of its components), otherwise, we move the point* x *in whichever components are within* r *from* $\partial\mathbb{X}$ *to be a distance* r *away from* $\partial\mathbb{X}$*. Consequently, for all* $0 < r < \frac{1}{2}$ *and for all* $x \in \mathbb{X}$*,* $\mathcal{B}^r(x)$ *has a bona fide radius of* r*, as the center* $\beta^r(x)$ *is smaller or equal to* r *away from* $\partial\mathbb{X}$*.*

For the r-box estimator, we have the following convergence results. The theorem below discusses the convergence rate of the average Wasserstein distance between the unknown kernel evaluated at any point and its estimator, when the radius r is chosen optimally with respect to the data sample M. Section 4.2 is dedicated to its proof.

Theorem 7 *Under Assumptions 2 and 3, choose r as follows*

$$r \sim \begin{cases} M^{-\frac{1}{d_{\mathbb{X}}+2}}, & d_{\mathbb{Y}} = 1, 2 \\ M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}, & d_{\mathbb{Y}} \geq 3. \end{cases}$$

Then, there is a constant $C > 0$ (which depends only on $d_{\mathbb{X}}, d_{\mathbb{Y}}, L, \underline{c}$), such that, for all probability distribution $\nu \in \mathcal{P}(\mathbb{X})$, we have

$$\mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{P}_x^r\right) \nu(\mathrm{d}x)\right] \leq \sup_{x \in \mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{P}^r\right)\right] \leq C \times \begin{cases} M^{-\frac{1}{d_{\mathbb{X}}+2}}, & d_{\mathbb{Y}} = 1, \\ M^{-\frac{1}{d_{\mathbb{X}}+2}} \ln(M), & d_{\mathbb{Y}} = 2, \\ M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}, & d_{\mathbb{Y}} \geq 3. \end{cases} \tag{3}$$

Next, we bound the associated variance whose proof is postponed to Section 4.3.

Theorem 8 *Under Assumptions 3, consider $r \in (0, \frac{1}{2}]$. Let $\nu \in \mathcal{P}(\mathbb{X})$ be dominated by $\lambda_{\mathbb{X}}$ with a constant $\overline{C} > 0$. Then,*

$$\mathrm{Var}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{P}_x^r\right) \mathrm{d}\nu(x)\right] \leq \frac{4^{d_{\mathbb{X}}+1}\overline{C}^2}{\underline{c}^2(M+1)}.$$

2.3 Results on k-nearest-neighbor estimator

Here, we focus in the second estimator – the k-nearest-neighbor estimator, defined as follows.

Definition 9 *Let $k \geq 1$ an integer. The k-nearest-neighbor estimator for P is defined by*

$$\begin{aligned} \check{P}^k : \mathbb{X} &\to \mathcal{P}(\mathbb{Y}) \\ x &\mapsto \check{P}_x^k := \hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)}, \end{aligned}$$

where, for any integer $M \geq 1$ and any $\mathsf{D}_{\mathbb{X}} \in \mathbb{X}^M$, $\mathcal{N}^{k,\mathsf{D}_{\mathbb{X}}}(x)$ contains (exactly) k points of $\mathsf{D}_{\mathbb{X}}$ which are closest to x, namely

$$\mathcal{N}^{k,\mathsf{D}_{\mathbb{X}}}(x) := \left\{x' \in \mathsf{D}_{\mathbb{X}} \,\middle|\, \|x - x'\|_\infty \text{ is among the } k\text{-smallest of } (\|x - x'\|_\infty)_{x' \in \mathsf{D}_{\mathbb{X}}}\right\},$$

Here, in case of a tie when choosing the k-th smallest, we break the tie randomly with uniform probability.

We have the following analogs of the convergence results (Theorems 7 and 8) for the k-nearest-neighbor estimator. The proofs are postponed to Section 4.4 and Section 4.5, respectively.

Theorem 10 *Under Assumptions 2 and 3, and choosing k as*

$$k \sim \begin{cases} M^{\frac{2}{d_{\mathbb{X}}+2}}, & d_{\mathbb{Y}} = 1, 2, \\ M^{\frac{d_{\mathbb{Y}}}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}, & d_{\mathbb{Y}} \geq 3, \end{cases}$$

there is a constant $C > 0$ (which depends only on $d_{\mathbb{X}}, d_{\mathbb{Y}}, L, \underline{c}$), such that, for all probability distribution $\nu \in \mathcal{P}(\mathbb{X})$, we have

$$\mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right) \nu(\mathrm{d}x)\right] \leq \sup_{x \in \mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \check{P}_x^k\right)\right] \leq C \times \begin{cases} M^{-\frac{1}{d_{\mathbb{X}}+2}}, & d_{\mathbb{Y}} = 1, \\ M^{-\frac{1}{d_{\mathbb{X}}+2}} \ln M, & d_{\mathbb{Y}} = 2, \\ M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}, & d_{\mathbb{Y}} \geq 3. \end{cases} \tag{4}$$

Theorem 11 *Under Assumptions 3, for any $\nu \in \mathcal{P}(\mathbb{X})$, we have*

$$\mathrm{Var}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right) \nu(\mathrm{d}x)\right] \leq \frac{1}{k}. \tag{5}$$

Moreover, if ν is dominated by $\lambda_{\mathbb{X}}$ with a constant $\overline{C} > 0$, then

$$\begin{aligned}&\mathrm{Var}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right) \nu(\mathrm{d}x)\right] \\ &\leq \frac{2^{2d_{\mathbb{X}}+1}\overline{C}^2 M}{\underline{c}^2 k^2}\left(\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}}+\frac{k}{M-1}\right)^2+\frac{\sqrt{2\pi}}{\sqrt{M-1}}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}}+\frac{k}{M-1}\right)+\frac{4}{M-1}\right).\end{aligned}$$

With k chosen as in Theorem 10, this reduces to

$$\mathrm{Var}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right) \nu(\mathrm{d}x)\right] \lesssim \begin{cases} M^{-\frac{2(2\vee d_{\mathbb{Y}})}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}\ln(M), & 2 \vee d_{\mathbb{Y}} \leq d_{\mathbb{X}}, \\ M^{-1}, & 2 \vee d_{\mathbb{Y}} > d_{\mathbb{X}}. \end{cases}$$

2.4 Comments on the convergence rate

This section gathers several comments on the convergence results we have developed in Sections 2.2 and 2.3.

2.4.1 On the convergence rate

We first comment on the expectations in Theorem 7 and 10.

Sharpness of the bounds. Currently, we cannot establish the sharpness of the convergence rates in Theorems 7 and 10. However, we can compare our results to established results in similar settings. For $d_{\mathbb{Y}} = 1$, we may compare it to the optimal rate of non-parametric regression of a Lipschitz continuous function. It is shown in Stone (1982) that the optimal rate is $M^{-\frac{1}{d_{\mathbb{X}}+2}}$, the same as in Theorems 7 and 10 when $d_{\mathbb{Y}} = 1$. For $d_{\mathbb{Y}} \geq 3$, as noted in Backhoff et al. (2022), we may compare to the Wasserstein convergence rate of empirical measure in the estimation of a static distribution on $\mathbb{R}^{d_{\mathbb{X}}+d_{\mathbb{Y}}}$. We refer to Fournier and Guillin (2015) for the optimal rate, which coincides with those in Theorems 7 and 10.

Error components. We discuss the composition of our upper bound on the expected average Wasserstein error by dissecting the proof of Theorem 7 and 10. In the proofs, we decompose the expected average errors into two components: approximation error and estimation error. The approximation error occurs when treating $P_{x'}$ as equal to P_x when x' is close to the query point x, leading to an error of size $L\|x - x'\|_\infty$. The estimation error is associated with the Wasserstein error of empirical measure under i.i.d. sampling (see (21)). From Definitions 5 and 9, the r-box estimator effectively manages the approximation error but struggles with controlling the estimation error, whereas the k-nearest-neighbor estimator exhibits the opposite behavior.

Explicit bounds. We primarily focus on analyzing the convergence rates of the r-box and k-nearest-neighbor estimators as $M \to \infty$. Therefore, within the proofs of these results, we track only the rates (and ignore various constant coefficients). If more explicit bounds are preferred, intermediate results such as (23), or (27) could be good starting points for computing them. Additionally, in Section A, we provide a numerical illustration that highlights the impact of the Lipschitz constant L and the choices of r and k on the bounds.

2.4.2 On the fluctuation

We next discuss the variances studied in Theorems 8 and 11. In Appendix D, we also include results derived from the Azuma-Hoeffding inequality (e.g., (Wainwright, 2019, Corollary 2.20)), though they provide rougher rates.

Condition that ν is dominated by $\lambda_{\mathbb{X}}$. In Theorems 8 and 11, we assume that the ν is dominated by $\lambda_{\mathbb{X}}$. This assumption is somewhat necessary. To illustrate, let us examine the non-parametric regression problem under a comparable scenario. We consider a fixed query point. In this context, the central limit theorem for k-nearest-neighbor estimator is well-established, and the normalizing rate is $k^{-\frac{1}{2}}$ (cf. (Biau and Devroye, 2015, Theorem 14.2)). This suggests that the rate in (5) is sharp. For the r-box estimator, we believe that a supporting example can be constructed where ν is highly concentrated. On the other hand, we conjecture that if $\xi \sim \nu$, the variance could potentially attain the order of M^{-1}. For a pertinent result, we direct the reader to (Backhoff et al., 2022, Theorem 1.7).

Sharpness of the bounds. Regarding the variance in Theorem 8, it is upper bounded by the commonly observed order of M^{-1}. We believe that this rate is sharp, though we do not have a proof at this time. As for Theorem 11, the variance is subject to a rougher rate when $2 \vee d_{\mathbb{Y}} \leq d_{\mathbb{X}}$. We, however, conjecture that this variance attains the order of M^{-1} as long as ν is dominated by $\lambda_{\mathbb{X}}$.

2.5 Towards implementation with neural networks

In light of recent practices in machine learning, during the learning of P, we may combine the r-box method or k-nearest-neighbor method into the training of certain parameterized model. To this end we let

$$\begin{aligned} \tilde{P} : \mathbb{T} \times \mathbb{X} &\to \mathcal{P}(\mathbb{Y}) \\ (\theta, x) &\mapsto \tilde{P}^{\theta}_{x} \end{aligned}$$

be a parameterized model (e.g., a neural network), where $\mathbb{T}$ is the parameter space and $\theta \in \mathbb{T}$ is the parameter to be optimized over. Given an integer $N \geq 1$, we may train $\tilde{P}^{\theta}$ on a set of query points $\mathcal{Q} = (\tilde{X}_n)_{n=1}^N$ satisfying the assumption below.

Assumption 12 *The query points $\mathcal{Q} = \{(\tilde{X}_n)\}_{n=1}^N$ are i.i.d. with uniform distribution over $\mathbb{X}$, and are independent of the data points $\mathcal{D} = \{(X_m, Y_m)\}_{m=1}^M$.*

We propose the training objectives below

$$\arg\min_{\theta \in \mathbb{T}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{W}\left(\hat{P}^{r}_{\tilde{X}_n}, \tilde{P}^{\theta}_{\tilde{X}_n}\right) \quad \text{or} \quad \arg\min_{\theta \in \mathbb{T}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{W}\left(\check{P}^{k}_{\tilde{X}_n}, \tilde{P}^{\theta}_{\tilde{X}_n}\right), \tag{6}$$

that is, minimize the mean of 1-Wasserstein errors between the parametrized model and the empirical r-box (or k-nearest-neighbour) approximation of the conditional distribution at the location of the random query points.

The following proposition together with Theorem 7 or Theorem 10 justifies using the objectives in (6). It is valid for any estimator for P that satisfies the bounds in (3) or (4). Centered on appropriate Lipschitz continuity conditions, the proposition offers insights into the worst-case performance guarantees. The proof is deferred to Section 4.6. We also refer to Altekrüger et al. (2023) for a worst-case performance guarantee for conditional generative models, which is contingent upon Lipschitz continuity. For related practical approaches, we refer to, for example, Nguyen et al. (2024) and the references therein. In contrast, similar guarantees for the r-box and k-nearest-neighbor estimators are more elusive due to their inherently piece-wise constant nature.

Proposition 13 *Suppose Assumptions 2, 3, and 12 hold. Let $\overline{P}$ of P be an estimator constructed using the data points $\mathcal{D}$ only. Consider a training procedure that produces a (random) $\Theta = \Theta(\mathcal{D}, \mathcal{Q})$ satisfying*

$$\sup_{x,x' \in \mathbb{X}} \frac{\mathcal{W}(\tilde{P}_x^\Theta, \tilde{P}_{x'}^\Theta)}{\|x - x'\|_\infty} \leq L^\Theta \tag{7}$$

for some (random) $L^\Theta > 0$. Then,

$$\begin{aligned} \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}(P_x, \tilde{P}_x^\Theta)\,\mathrm{d}x\right] \leq \mathbb{E}\left[(L + L^\Theta)\mathcal{W}\left(\lambda_{\mathbb{X}}, \frac{1}{N}\sum_{n=1}^N \delta_{\tilde{X}_n}\right)\right] \\ + \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}(P_x, \overline{P}_x)\,\mathrm{d}x\right] + \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \mathcal{W}\left(\overline{P}_{\tilde{X}_n}, \tilde{P}_{\tilde{X}_n}^\Theta\right)\right]. \end{aligned} \tag{8}$$

Moreover, with probability 1*,*

$$\sup_{x \in \mathbb{X}} \mathcal{W}\left(P_x, \tilde{P}_x^\Theta\right) \leq (d_{\mathbb{X}} + 1)^{\frac{1}{d_{\mathbb{X}}+1}} (L + L^\Theta)^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+1}} \left(\int_{\mathbb{X}} \mathcal{W}(P_x, \tilde{P}_x^\Theta)\,\mathrm{d}x\right)^{\frac{1}{d_{\mathbb{X}}+1}}. \tag{9}$$

Remark 14 *Assuming $L^\Theta \leq \overline{L}$ for some (deterministic) $\overline{L} > 0$, by* (9) *and Jensen's inequality, we have*

$$\mathbb{E}\left[\sup_{x \in \mathbb{X}} \mathcal{W}\left(P_x, \tilde{P}_x^\Theta\right)\right] \leq (d_{\mathbb{X}} + 1)^{\frac{1}{d_{\mathbb{X}}+1}} (L + \overline{L})^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+1}} \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}(P_x, \tilde{P}_x^\Theta)\,\mathrm{d}x\right]^{\frac{1}{d_{\mathbb{X}}+1}}.$$

This together with (8) *provides a worst-case performance guarantee for $\tilde{P}^\Theta$.*

Remark 15 *Proposition 13 along with Remark 14 provides insights into the worst-case performance guarantees, but more analysis is needed. Specifically, understanding the magnitude of L^Θ and $\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \mathcal{W}(\overline{P}_{\tilde{X}_n}, \tilde{P}_{\tilde{X}_n}^\Theta)\right]$ requires deeper knowledge of the training processes for $\tilde{P}^\Theta$, which are currently not well understood in the extant literature. Alternatively, in the hypothetical case where $\tilde{P}^\Theta = P$, L^Θ would match L as specified in Assumption 2, and $\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N \mathcal{W}\left(\overline{P}_{\tilde{X}_n}, \tilde{P}_{\tilde{X}_n}^\Theta\right)\right]$ would obey Theorem 7 or 10. However, practical applications must also consider the universal approximation capability of $\tilde{P}^\theta$. To the best of our knowledge, research on universal approximation with regularity constraints remains relatively limited. For a somewhat related study, we refer to Hong and Kratsios (2024) who explore the approximation of real-valued functions under Lipschitz continuity constraints.*

3. Implementation with neural networks

Let $\mathbb{X}$ and $\mathbb{Y}$ be equipped with $\|\cdot\|_1$. Following the discussion in Section 2.5, we let $\tilde{P}^\theta : \mathbb{X} \to \mathcal{P}(\mathbb{Y})$ be parameterized by a neural network and develop an algorithm that trains $\tilde{P}^\theta$ based on k-nearest-neighbor estimator. The k-nearest-neighbor estimator $\check{P}^k$ is preferred as $\check{P}_x^k$ consistently outputs k atoms. This regularity greatly facilities implementation. For instance, it enables the use of 3D tensors during Sinkhorn iterations to enhance execution speed (see Section 3.1.2 later). We refer also to the sparsity part of Section 5.2 for another component that necessitates the aforementioned regularity of $\check{P}^k$. These components would not be feasible with the r-box estimator $\hat{P}^r$, as $\hat{P}_x^r$

produces an undetermined number of atoms. Furthermore, there is a concern that in some realizations, $\hat{P}^r_x$ at certain x may contain too few data points, potentially leading $\tilde{P}^\Theta_x$ to exhibit unrealistic concentration.

We next provide some motivation for this implementation. For clarity, we refer to the r-box estimator and the k-nearest-neighbor estimator as raw estimators. Additionally, we refer to $\tilde{P}^\Theta$, once trained, as the neural estimator. While raw estimators are adequate for estimating P on their own, they are piece-wise constant in x by design. On the other hand, a neural estimator is continuous in x. This continuity provides a performance guarantee in $\sup \mathcal{W}$ distance, as outlined in Proposition 13 and the following remark. Moreover, the neural estimator inherently possesses gradient information. As discussed in the introduction, this feature renders the neural estimators useful in downstream contexts where gradient information is important, e.g., when performing model-based reinforcement learning.

We construct $\tilde{P}^\theta$ such that it maps $x \in \mathbb{X}$ to atoms in $\mathbb{Y}$ with equal probabilities. For the related universal approximation theorems, we refer to Kratsios (2023); Acciaio et al. (2024). We represent these atoms with a vector with N_{atom} entries denoted by $y^\theta(x) = (y^\theta_1(x), \dots, y^\theta_{N_{\text{atom}}}(x)) \in \mathbb{Y}^{N_{\text{atom}}}$, where $N_{\text{atom}} \in \mathbb{N}$ is chosen by the user. In our implementation, we set $N_{\text{atom}} = k$. To be precise, we construct $\tilde{P}^\theta$ such that

$$\tilde{P}^\theta_x = \frac{1}{N_{\text{atom}}} \sum_{j=1}^{N_{\text{atom}}} \delta_{y^\theta_j(x)}, \quad x \in \mathbb{N}. \tag{10}$$

This is known as the Lagrangian discretization (see (Peyré and Cuturi, 2019, Section 9)). In Algorithm 1, we present a high level description of our implementation of training $\tilde{P}^\theta$ based on the raw k-nearest-neighbor estimator.

Algorithm 1 Deep learning conditional distribution in conjunction with k-NN estimator

Input: data $\{(X_m, Y_m)\}_{m=1}^M$ valued in $\mathbb{R}^{d_\mathbb{X}} \times \mathbb{R}^{d_\mathbb{Y}}$, neural estimator $\tilde{P}^\theta$ represented by $y^\theta(x)$ as elaborated in (10), parameters such as $k, N_{\text{atoms}}, N_{\text{batch}} \in \mathbb{N}_+$, and learning rate η_θ

Output: trained parameter Θ for the neural estimator

repeat
  for $n = 1, \dots, N_{\text{batch}}$ **do**
    generate a query point $\tilde{X}_n \sim \text{Uniform}(\mathbb{X})$
    find the k nearest neighbors of $\tilde{X}_n$ from data $(X_m)_{m=1}^M$ and collect accordingly $(\tilde{Y}_{n,i})_{i=1}^k$
  end for
  compute with Sinkhorn algorithm ($\mathbb{Y}$ is equipped with $\|\cdot\|_1$)

$$L[\theta] := \sum_{n=1}^{N_{\text{batch}}} \mathcal{W}\left(\frac{1}{k}\sum_{i=1}^{k} \delta_{\tilde{Y}_{n,i}}, \frac{1}{N_{\text{atom}}} \sum_{j=1}^{N_{\text{atom}}} \delta_{y^\theta_j(\tilde{X}_n)}\right) \tag{11}$$

  update $\theta \leftarrow \theta - \eta_\theta \nabla_\theta L[\theta]$
until Convergence
return $\Theta = \theta$

3.1 Overview of key components

In this section, we outline the three key components of our implementation. Each of these components addresses a specific issue:

- Managing the computational cost arising from the nearest neighbors search.
- Implementing gradient descent after computing $\mathcal{W}$.

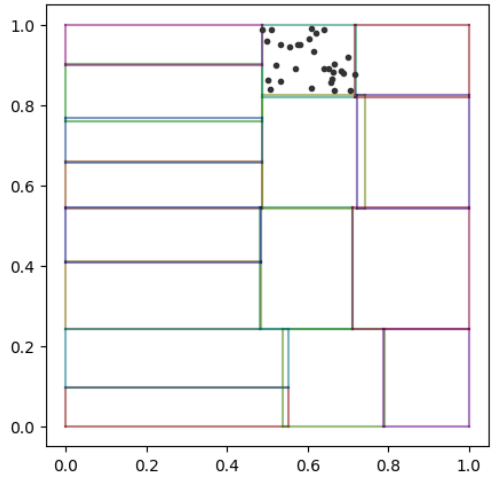


Figure 1: *An instance of RBSP in* $[0,1]^2$.

The 2D unit box is partitioned into 16 rectangles based on 500 samples from Uniform($[0,1]$). Note that the overlap between the bounding rectangles is intentionally maintained. Each partitioning is performed along an axis selected at random, dividing the samples within the pre-partitioned rectangle according to a random ratio drawn from Uniform($[0.45, 0.55]$). The edge ratio for mandatory bisecting along the longest edge is 5. If this ratio is exceeded, partitioning along the longest edge is enforced. The black dots represent samples within the respective rectangle.

- Selecting an appropriate Lipschitz constant for the neural estimator, preferably at a local level.

Further details and ablation analysis on these three components can be found in Section 5.

3.1.1 Approximate Nearest Neighbors Search with Random Binary Space Partitioning (ANNS-RBSP)

Given a query point, performing an exact search for its k-nearest-neighbor requires $O(M)$ operations. While a single search is not overly demanding, executing multiple searches as outlined in Algorithm 1 can result in significant computational time, even when leveraging GPU-accelerated parallel computing. To address this, we use ANNS-RBSP as a more cost-effective alternative. Prior to searching, we sort $(X_m)_{m=1}^M$ along each axis and record the order of indices. During the search, the data is divided into smaller subsets by repeatedly applying bisection on these sorted indices, with a random bisecting ratio, on a randomly chosen axis. Furthermore, we apply a restriction that mandates bisection along the longest edge of a rectangle when the edge ratio exceeds certain value (a hyper-parameter of the model). We record the bounding rectangle for each subset created through this partitioning process. Once partitioning is complete, we generate a small batch of query points within each rectangle and identify the k nearest neighbors for each query point within that same rectangle. For a visual representation of ANNS-BSP, we refer to Figure 1. Leveraging the sorted indices, we can reapply this partitioning method during every training episode without much computational cost. We refer to Section 5.1 for additional details. There are similar ideas in the extant literature (cf. Hajebi et al. (2011); Ram and Sinha (2019); Li et al. (2020a)). Given the substantial differences in our setting, however, we conduct further empirical analysis in Section 5.1 to showcase the advantage of our approach against exact search.

3.1.2 Computing $\mathcal{W}$ for gradient descent

The following discussion pertains to the computation of (11), with the subsequent gradient descent in consideration. For simplicity, let us focus on the summand and reduce the problem to the following minimization. Let $(\tilde{y}_1, \dots, \tilde{y}_k) \in \mathbb{Y}^k$ be fixed, we aim to find

$$\operatorname*{arg\,min}_{y \in \mathbb{Y}^n} \mathcal{W}\left(\frac{1}{k}\sum_{i=1}^k \delta_{\tilde{y}_i}, \frac{1}{n}\sum_{j=1}^n \delta_{y_j} \right). \tag{12}$$

The criterion in (12) is convex as $\mathcal{W}$ is convex in both arguments (cf. (Villani, 2008, Theorem 4.8)). To solve (12), as is standard, we cast it into a discrete optimal transport problem. To do so, first introduce the $(k \times n)$-cost matrix C_y, where $\mathsf{C}_{y,ij} := \|\tilde{y}_i - y_j\|_1$. As the criterion in (12) has uniform weights on the atoms, we next aim to solve the problem

$$\begin{aligned} &\operatorname*{arg\,min}_{\mathsf{T} \in [0,1]^{k\times n}} \left\{ \varphi_y(\mathsf{T}) := \sum_{(i,j)\in\{1,\dots,k\}\times\{1,\dots,n\}} \mathsf{T}_{ij}\mathsf{C}_{y,ij} \right\} \\ &\text{subject to} \quad \sum_{j=1}^n \mathsf{T}_{ij} = \frac{1}{k},\ i = 1,\dots,k \quad \text{and} \quad \sum_{i=1}^k \mathsf{T}_{ij} = \frac{1}{n},\ j = 1,\dots,n. \end{aligned} \tag{13}$$

Let T^*_y be an optimal transport plan that solves (13) for y fixed. Taking derivative of $y \mapsto \varphi_y(\cdot)$ yields

$$\partial_{y_j} \varphi_y(\mathsf{T})\big|_{\mathsf{T}=\mathsf{T}^*_y} = \sum_{i\in\{1,\dots,k\}} \mathsf{T}^*_{y,ij}\, \partial_{y_j} \|\tilde{y}_i - y_j\|_1, \quad j = 1,\dots,n. \tag{14}$$

This gradient is in general not the gradient corresponding to (12), as T^*_y depends on y, while (14) excludes such dependence. Nevertheless, it is still viable to update y using the gradient descent that employs the partial gradient specified in (14). To justify this update rule, first consider $y' \in \mathbb{Y}$ satisfying $\varphi_{y'}(\mathsf{T}^*_y) \le \varphi_y(\mathsf{T}^*_y)$, then observe that

$$\mathcal{W}\left(\frac{1}{k}\sum_{i=1}^k \delta_{\tilde{y}_i}, \frac{1}{n}\sum_{j=1}^n \delta_{y'_j} \right) \le \varphi_{y'}(\mathsf{T}^*_y) \le \varphi_y(\mathsf{T}^*_y) = \mathcal{W}\left(\frac{1}{k}\sum_{i=1}^k \delta_{\tilde{y}_i}, \frac{1}{n}\sum_{j=1}^n \delta_{y_j} \right).$$

This inequality is strict if $\varphi_{y'}(\mathsf{T}^*_y) < \varphi_y(\mathsf{T}^*_y)$. We refer to (Peyré and Cuturi, 2019, Section 9.1) and the reference therein for related discussions.

The Sinkhorn algorithm, which adds an entropy regularization, is a widely-used algorithm for approximating the solution to (13). Specifically, here, it is an iterative scheme that approximately solves the following regularized problem, subject to the constraints in (13),

$$\operatorname*{arg\,min}_{\mathsf{T}^\epsilon \in [0,1]^{k\times n}} \left\{ \sum_{i,j\in\{1,\dots,k\}\times\{i,\dots,n\}} \mathsf{T}^\epsilon_{ij}\mathsf{C}_{ij} + \epsilon \sum_{i,j\in\{1,\dots,k\}\times\{i,\dots,n\}} \mathsf{T}^\epsilon_{ij}(\log \mathsf{T}_{ij} - 1) \right\}, \tag{15}$$

where $\epsilon > 0$ is a hyper-parameter, and should not be confused with the ε used elsewhere. We refer to Section 5.2 for further details. We also refer to (Peyré and Cuturi, 2019, Section 4) and the reference therein for convergence analysis of the Sinkhorn algorithm. It is well known that the regularization term in (15) is related to the entropy of a discrete random variable. Larger values of ϵ encourages the regularized optimal transport plan to be more diffusive. That is, for larger values

of ϵ, the mass from each y_j is distributed more evenly across all $\tilde{y}_i$'s. Performing gradient descent along the direction in (14) tends to pull y_j's towards the median of the $\tilde{y}_i$'s, as we are equipping $\mathbb{Y}$ with the norm $\|\cdot\|_1$. Conversely, small values of ϵ often leads to instability, resulting in NaN loss/gradient. To help with these issues, we implement the Sinkhorn algorithm after normalizing the cost matrix. Additionally, we use a large ϵ (e.g., 1) in the first few training episodes, then switch to a smaller ϵ (e.g., 0.1) in later episodes. Furthermore, we impose sparsity on the transport plan by manually setting the smaller entries of the transport plan to 0. The specific detailed configurations and related ablation analysis are provided in Section 5.2 and Appendix C.

3.1.3 Network structure that induces locally adaptive Lipschitz continuity

As previously discussed, it is desirable for the neural estimator to exhibit certain Lipschitz continuity. In practice, however, determining an appropriate Lipschitz constant for training the neural estimator $\tilde{P}^\theta$ is challenging, largely because understanding the true Lipschitz continuity of P (if it exists) in a data-driven manner is very challenging. Additionally, the estimate provided in Proposition 13 is probabilistic. Fortunately, a specific network structure allows the neural estimator, when properly trained, to exhibit locally adaptive Lipschitz continuity. Subsequently, we provide a high-level overview of this network structure. Further detailed configurations and ablation analysis are presented in Section 5.3 and Appendix C.

Consider a fully connected feed-forward neural network with equal width hidden layers and layer-wise residual connection (He et al. (2016)). Let N_{neuron} denote the width of the hidden layers. For activation, we use Exponential Linear Unit (ELU) function (Clevert et al. (2016)), denoted by σ. For hidden layers, we employ the convex potential layer introduced in Meunier et al. (2022),

$$\mathsf{x}_{\text{out}} = \mathsf{x}_{\text{in}} - \|\mathsf{W}\|_2^{-1}\mathsf{W}^\mathsf{T}\sigma\left(\mathsf{W}\mathsf{x}_{\text{in}} + \mathsf{b}\right). \tag{16}$$

By (Meunier et al., 2022, Proposition 3), the convex potential layer is 1-Lipschitz continuous in $\|\cdot\|_2$ sense. For the input layer, with a slight abuse of notation, we use

$$\mathsf{x}_{\text{out}} = N_{\text{neuron}}^{-1}\text{diag}(|\mathsf{W}|_1^{-1} \wedge 1)\sigma\left(\mathsf{W}\mathsf{x}_{\text{in}} + \mathsf{b}\right), \tag{17}$$

where $|\mathsf{W}|_1$ computes the absolute sum of each row of the weight matrix to form a vector of size N_{neuron}, the reciprocal and $\cdot \wedge 1$ are applied entry-wise, and diag produces a diagonal square matrix based on the input vector. In short, the normalization in (17) is only applied to the rows of W with ℓ_1-norm exceeding 1. Consequently, the input layer is 1-Lipschitz continuous in $\|\cdot\|_1$ sense. A similar treatment is used for the output layer but without activation,

$$\mathsf{x}_{\text{out}} = L\, d_{\mathbb{Y}}^{-1}\text{diag}(|\mathsf{W}|_1^{-1} \wedge 1)\left(\mathsf{W}\mathsf{x}_{\text{in}} + \mathsf{b}\right). \tag{18}$$

where $L > 0$ is a hyper-parameter. The output represents atoms on $\mathbb{Y}$ with uniform weight, therefore, no N_{atom}^{-1} is required here.

The spectral norm $\|\mathsf{W}\|_2$ in (16), however, does not, in general, have an explicit expression. Following the implementation in Meunier et al. (2022), we approximate each $\|\mathsf{W}\|_2$ with power iteration. Power iterations are applied to all hidden layers simultaneously during training. To control the pace of iterations, we combine them with momentum-based updating. We refer to Algorithm 2 for the detailed implementation. Our implementation differs from that in Meunier et al. (2022), as Meunier et al. (2022) controls the frequency of updates but not the momentum. In a similar manner, for input and output layers, instead of calculating the row-wise ℓ_1-norm explicitly, we update them with the same momentum used in the hidden layers. Our numerical experiments consistently show that a small momentum value of $\tau = 10^{-3}$ effectively maintains

Algorithm 2 Power iteration with momentum for updating $\|\mathsf{W}\|_2$ estimate, applied to all convex potential layers simultaneously at every epoch during training

Input: weight matrix $\mathsf{W} \in \mathbb{R}^{d\times d}$ of a convex potential layer, previous estimate $\hat{h} \in \mathbb{R}$ and auxiliary vector $\hat{\mathsf{u}} \in \mathbb{R}^d$, momentum $\tau \in (0,1)$

Output: updated $\hat{h}$ and $\hat{\mathsf{u}}$, in particular, $\hat{h}$ will be used as a substitute of $\|\mathsf{W}\|_2$ in (16)

$\mathsf{v} \leftarrow \mathsf{W}\hat{\mathsf{u}}/\|\mathsf{W}\hat{\mathsf{u}}\|_2$
$\mathsf{u} \leftarrow \mathsf{W}^\mathsf{T}\mathsf{v}/\|\mathsf{W}^\mathsf{T}\mathsf{v}\|_2$
$h \leftarrow 2/\left(\sum_i(\mathsf{W}\mathsf{u}\cdot\mathsf{v})_i\right)^2$
} power iteration
$\hat{h} \leftarrow \tau\hat{h} + (1-\tau)h$
$\hat{\mathsf{u}} \leftarrow \tau\hat{\mathsf{u}} + (1-\tau)\mathsf{u}$
} momentum-based updating
return $\hat{h}$, $\hat{\mathsf{u}}$

adaptive continuity while maintaining a satisfactory accuracy. The impact of L in (18) and τ in Algorithm 2 is discussed in Section 5.3.

During training, due to the nature of our updating schemes, the normalizing constants do not achieve the values required for the layers to be 1-Lipschitz continuous. We hypothesize that this phenomenon leads to a balance that ultimately contributes to adaptive continuity: on one hand, the weights W stretch to fit (or overfit) the data, while on the other, normalization through iterative methods prevents the network from excessive oscillation. As shown in Section 5.3.2 and 5.3.3, the L value in (18) and the momentum τ in Algorithm 2 affect the performance significantly. For completeness, we also experiment with replacing (16) by fully connected feedforward layers similar to (17), with or without batch normalization (Ioffe and Szegedy (2015)) after affine transformation. This alternative, however, failed to produce satisfactory results.

3.2 Experiments with synthetic data

We consider data simulated from three different models. The first two have $d_\mathbb{X} = d_\mathbb{Y} = 1$, while the third has $d_\mathbb{X} = d_\mathbb{Y} = 3$. Here we no longer restrict $\mathbb{Y}$ to be the unit box, however, we still consider $\mathbb{X}$ to be a $d_\mathbb{X}$-dimensional unit box (not necessarily centered at the origin).

In Model 1 and 2, $X \sim \text{Uniform}([0,1])$. Model 1 is a mixture of two independent Gaussian random variables with mean and variance depending on x,

$$Y = \xi\Big(0.1\left(1+\cos(2\pi X)\right) + 0.12\left|1-\cos(2\pi X)\right|Z + 0.5\Big),$$

where $Z \sim \text{Normal}(0,1)$ and ξ is a Rademacher random variable independent of Z. For Model 2, we have

$$Y = 0.5\,\mathbb{1}_{[0,1)}(X) + 0.5\,U,$$

where $U \sim \text{Uniform}([0,1])$. The conditional distribution in Model 2 is intentionally designed to be discontinuous in the feature space. This choice was made to evaluate performance in the absence of the Lipschitz continuity stipulated in Assumption 2. Model 3 is also a mixture of two independent Gaussian random variables, constructed by considering $X \sim \text{Uniform}([-\frac{1}{2},\frac{1}{2}]^3)$ and treating X as a column vector (i.e., X take values in $\mathbb{R}^{3\times 1}$),

$$Y = \zeta\Big(\cos(\mathsf{A}X) + 0.1\cos(\mathbf{\Sigma}_X)W\Big) + (1-\zeta)\Big(\cos(\mathsf{A}'X) + 0.1\cos(\mathbf{\Sigma}'_X)W'\Big).$$

Above, the cos functions act on vector/matrix entrywise, $\mathsf{A} \in \mathbb{R}^{3\times 3}$, and $\mathbf{\Sigma}_x$ also takes value in $\mathbb{R}^{3\times 3}$. Each element of $\mathbf{\Sigma}_x$ is defined as $\mathsf{v}_{ij}x$ for some $\mathsf{v}_{ij} \in \mathbb{R}^{1\times 3}$. The entries of A and v_{ij} are drawn from

standard normal in advance and remain fixed throughout the experiment. The matrices A' and $\mathsf{\Sigma}'_x$ are similarly constructed. Furthermore, W and W' are independent three-dimensional standard normal r.v.s, while ζ represents the toss of a fair coin, independent of X, W, and W'.

For the purpose of comparison, two different network structures are examined. The first, termed LipNet, is illustrated in Section 3.1. The second, termed StdNet, is a fully connected feedforward network with layer-wise residual connections He et al. (2016), ReLU activation, and batch normalization immediately following each affine transformation, without specifically targeting Lipschitz continuity. With a hyper-parameter k for the k-nearest-neighbor estimator, which we specify later, each network contains 5 hidden layers with $2k$ neurons. These networks are trained using the Adam optimizer Kingma and Ba (2017) with a learning rate of 10^{-3}. For StdNet in Model 1 and 2, the learning rate is set to 0.01, as it leads to better performance. Other than the learning rates, StdNet and LipNet are trained with identical hyper-parameters across all models. We refer to Appendix C for a summary of hyper-parameters involved.

We generate 10^4 samples for Models 1 and 2. Given the convergence rate specified in Theorem 10, we note that the sample size are considered relatively small. For these two models, we chose $k = 100$ and utilized neural networks $\tilde{P}^\theta$ that output atoms of size $N_{\text{atom}} = k$. The choice of k is determined by a rule of thumb. In particular, our considerations include the magnitude of k suggested by Theorem 10 and the computational costs associated with the Sinkhorn iterations discussed in Section 3.1.2. The results under Model 1 and 2 are plotted in Figure 2, 3 and 4. Figure 2 provides a perspective on joint distributions, while Figure 3 and 4 focus on conditional CDFs across different x values.

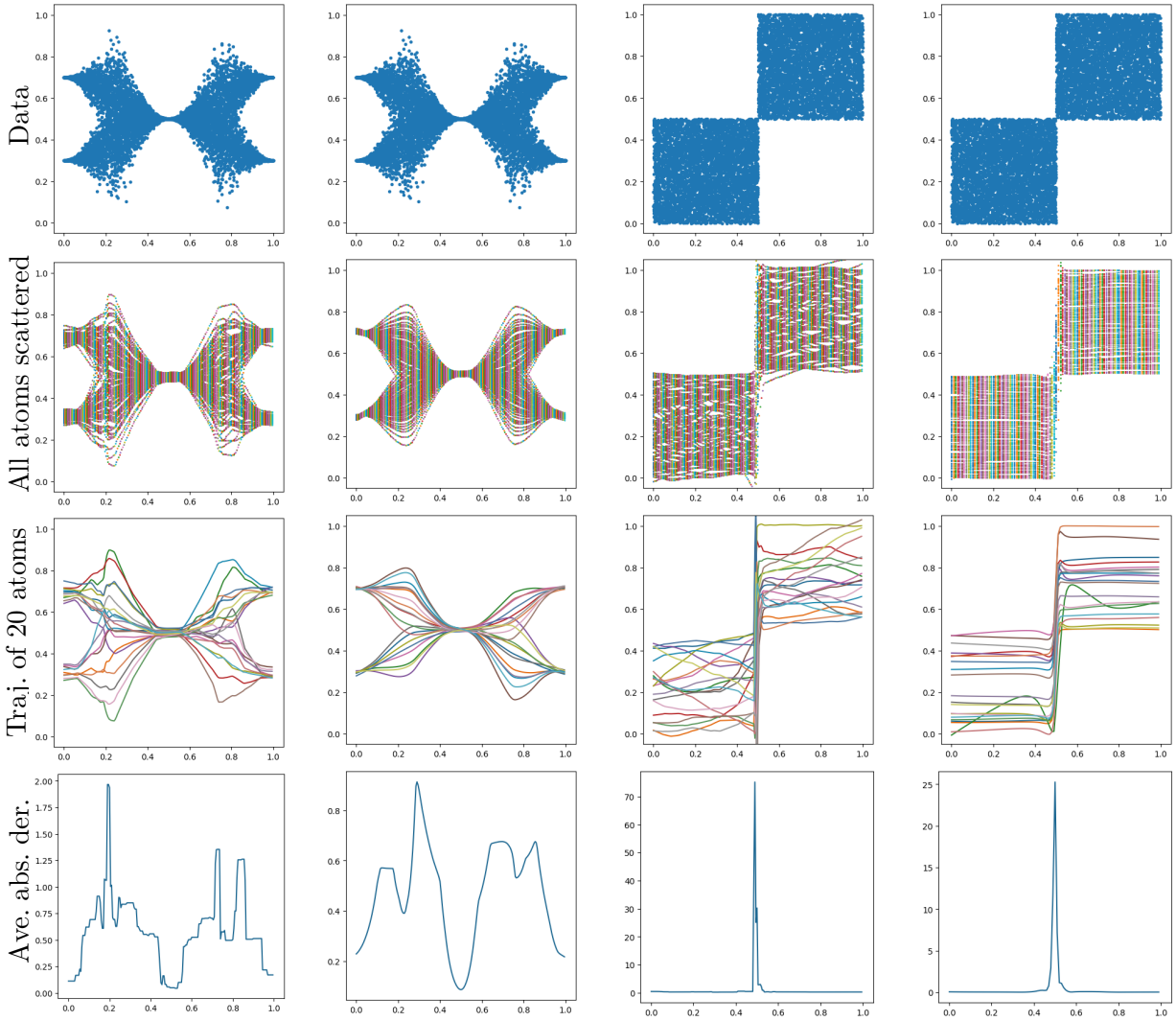
Figure 2 suggets that both StdNet and LipNet adequately recover the joint distribution. The LipNet's accuracy is, however, notably superior and produces smooth movements of atoms (as seen in the third row of Figure 2). Although further fine-tuning may provide slight improvements in StdNet's performance, StdNet will still not achieve the level of accuracy and smoothness observed in LipNet. The average absolute value of derivative of each atom (fourth row of Figure 2), makes it evident that LipNet demonstrates a capacity of automatically adapting to a suitable level of Lipschitz continuity locally. In particular, in Model 2, the atoms of LipNet respond promptly to jumps while remaining relatively stationary around values of x where the kernel is constant. We emphasize that LipNet is trained using the same hyper-parameters across Models 1, 2, and 3.

Figure 3 shows the estimated conditional distribution at different values of x. Figure 3 indicates that the raw k-nearest-neighbor estimator deviates frequently from the actual CDFs. This deviation of the raw k-nearest-neighbor estimator is expected, as it attempts to estimate an unknown CDF with only $k = 100$ samples given an x. Conversely, the neural estimator, especially the LipNet, appears to offer extra corrections even if they are trained based on the raw k-nearest-neighbor estimator. This could be attributed to neural estimators implicitly leveraging information beyond the immediate neighborhood.

Figure 4 compares the $\mathcal{W}$-distance between each estimator and the true conditional distribution at various values of x, using the following formula (see (Peyré and Cuturi, 2019, Remark 2.28)),

$$\mathcal{W}(F, G) = \int_{\mathbb{R}} \big|F(r) - G(r)\big| \, \mathrm{d}r, \tag{19}$$

where F and G are CDFs. This quantity can be accurately approximated with trapezoidal rule. In Model 1, the neural estimator generally outperforms the raw estimator with $k = 100$ across most values of x, even though the raw estimator is used for training the neural estimators. Furthermore, LipNet continues to outperform raw estimators with larger values of k – even though LipNet is trained with a raw estimator with $k = 100$. In Model 2, LipNet continues to demonstrate a superior performance, except when compared to the raw estimator with $k = 1,000$ at x distant from 0.5,



(a) *Model 1, StdNet* ***(b)*** *Model 1, LipNet* ***(c)*** *Model 2, StdNet* ***(d)*** *Model 2, LipNet*

Figure 2: *Various estimators under Model 1 and 2, joint distributions.*

The first row presents the data. Neural networks with different structures are trained on the same set of data for comparison. The second row shows scatter plots of atoms at various x values. The third row illustrates the evolution of 20 atoms as x varies. The final row presents the average of the derivative of each atom with respect to x, with a notable difference in the y axis scale.

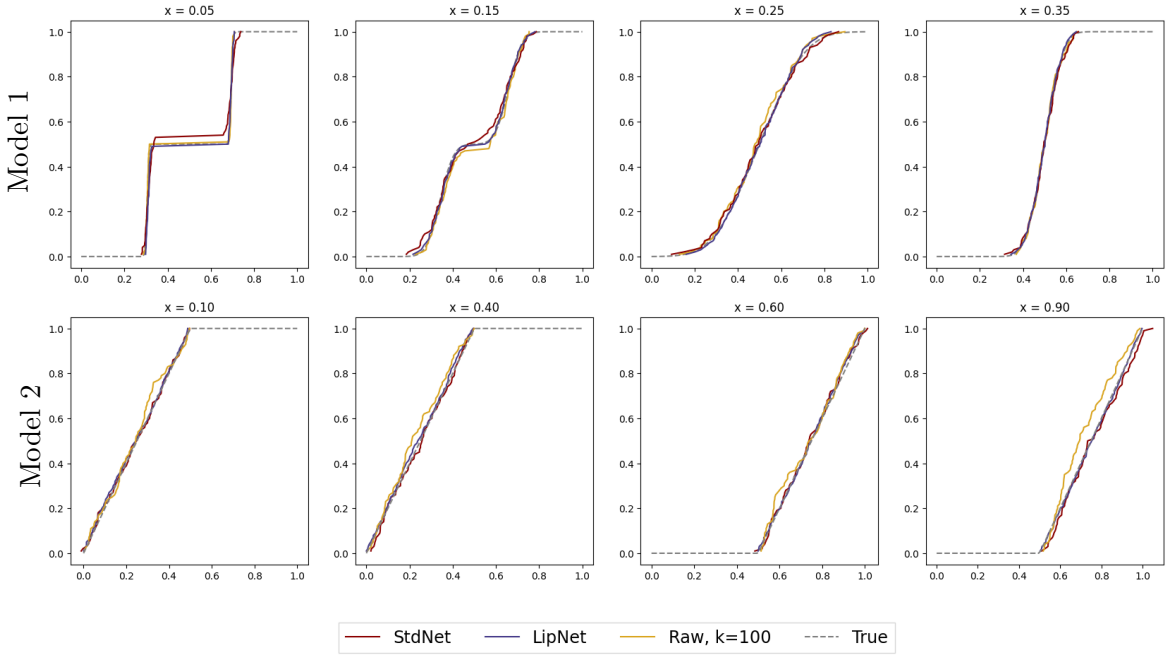


Figure 3: *Various estimators under Model 1 and 2, conditional CDFs.*

We compare the conditional CDFs at various values of x, derived from StdNet, LipNet, the raw k-nearest estimator with $k = 100$ (also used in the training of StdNet and LipNet), and the ground truth. The first row pertains to data set 1. The second row pertains to data set 2. Subfigure titles display the values of x.

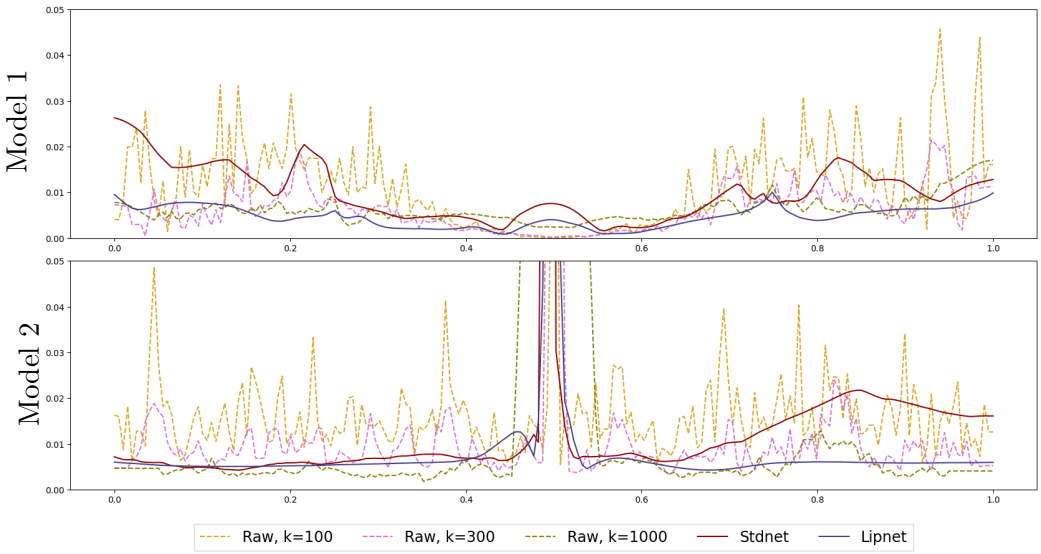


Figure 4: *Errors at different x's of various estimators under Model 1 and 2.*

We compute the $\mathcal{W}$-distance between estimators and the true conditional distribution at different x's. StdNet and LipNet are trained based on the raw k-nearest-neighbor estimator with $k = 100$.

the reason is that, here, the conditional distribution is piece-wise constant in x, which enhances the performance of the raw estimator at larger k values.

The aforementioned findings indicate superior performance by LipNet. We, however, recognize that improvements are not always guaranteed, as demonstrated in Figures 3 and 4.

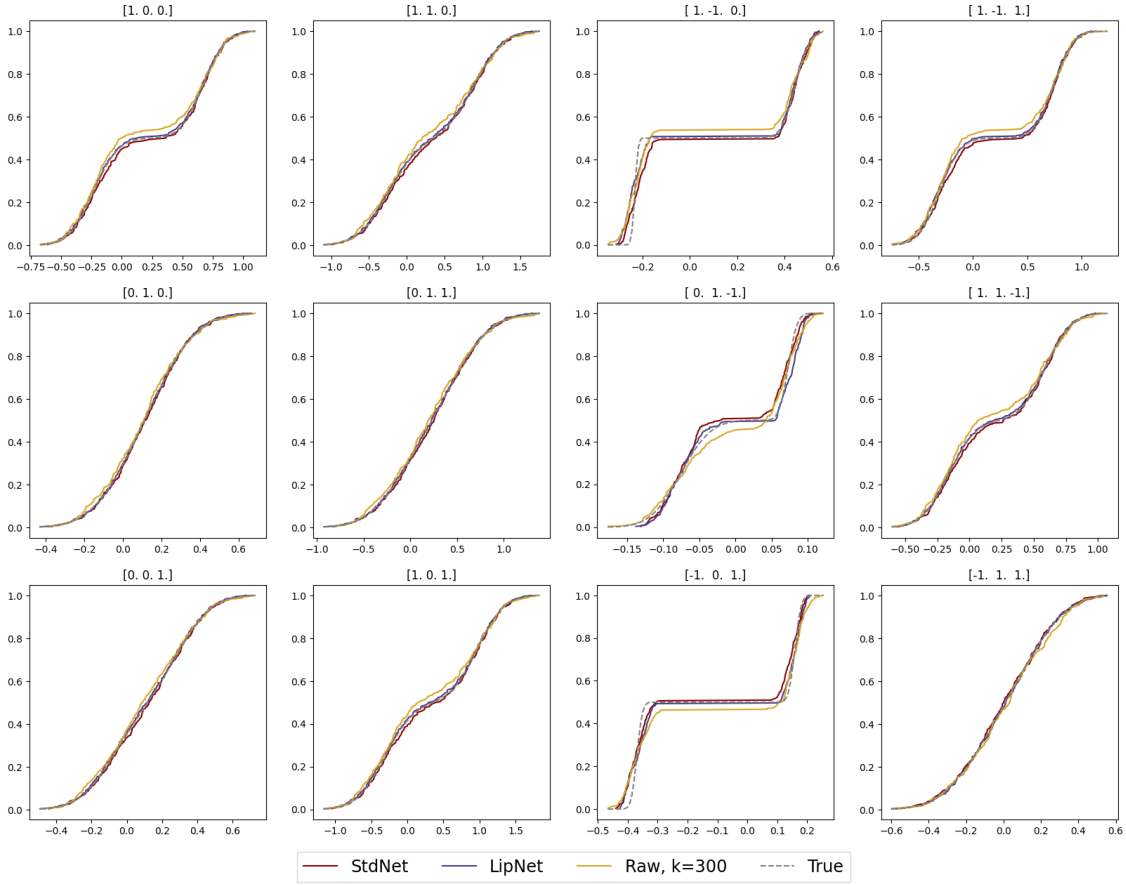
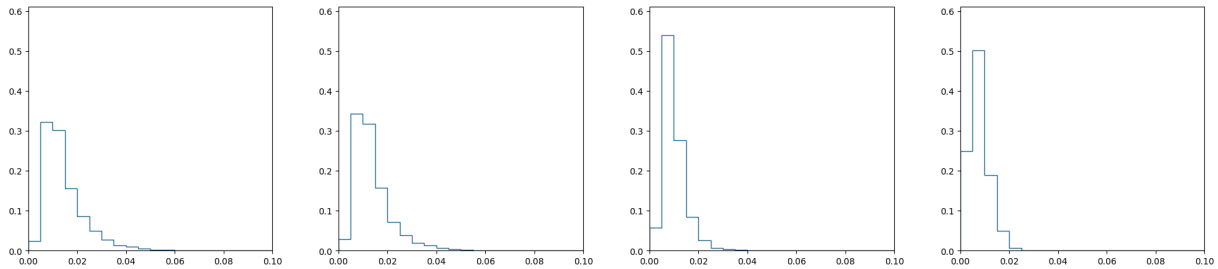


Figure 5: *Various estimators under Model 3, projections of conditional CDFs.*

We compare the projected conditional CDFs at $x = (0.12, -0.33, 0.1)$. The estimations are obtained from StdNet, LipNet, the raw estimator with $k = 300$ (also used in training StdNet and LipNet), and the ground truth. Subfigure titles display the vectors used for projection. Note the difference in the x axis scale.

For Model 3, we generate 10^6 samples and select $k = 300$. We train both neural estimators using Adam optimizer with a learning rate of 10^{-3}. Hyperparameters such as L in (18) and τ in Algorithm 2 are consistent with those used for Models 1 and 2. We refer to Table 4 for the detailed configuration.

In Figure 5, we visualize the outcomes in Model 3: the conditional CDFs at an arbitrarily chosen x are projected onto various vectors. We observe that the neural estimators considerably outperform the raw k-nearest-neighbor estimator, likely owing due to their implicit use of global information outside of the immediate neighbors during training. For further comparisons, we present additional figures in Appendix B: Figures 15, 16 and 17 feature the exact same neural estimators as shown in Figure 5, but with the raw k-nearest-neighbor estimators employing different k values, $k = 1,000,\ 3,000,\ 10,000$. Raw k-nearest-neighbor estimators with $k = 1,000,\ 3,000$ are superior to that with $k = 300$, while at $k = 10,000$, the accuracy begins to decline. Upon comparison,



Raw $k = 1,000$ *Raw* $k = 3,000$ *StdNet* *LipNet*

Figure 6: *Histogram of* $10,000$ *projected Wasserstein-1 errors.*
Each histogram consists of 20 uniformly positioned bins between 0 to 0.1. The errors of different estimators are computed with the same set of query points and projection vectors. Errors larger than 0.1 will be placed in the right-most bins. Note StdNet and LipNet are trained with $k = 300$.

the neural estimator trained with $k = 300$ consistently outperforms the raw k-nearest-neighbor estimators for all values of k.

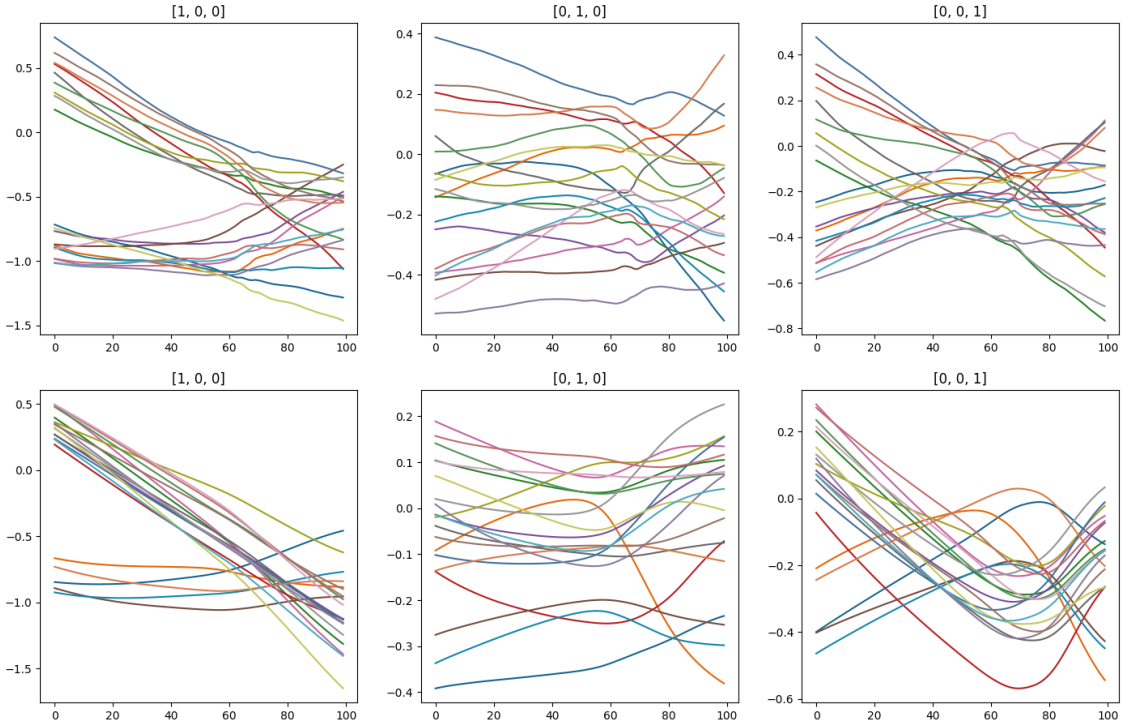


Figure 7: *LipNet under Model 3, projected trajectories of* 20 *atoms.*

We illustrate the projected trajectories of 20 atoms by evaluating LipNet at 100 evenly allocated points along the straight line that intersects the origin and $x = (0.12, -0.33, 0.1)$, situated within $[0, 1]^3$. The x-axis denotes the specific points along the line, consistent across all subfigures. Subfigure titles display the vectors used for projection. Note the difference in the y axis scale.

For a more comprehensive comparison, we randomly select $10,000$ query points. For each query point, we randomly generate a vector in $\mathbb{R}^3$, normalized under $\|\cdot\|_1$, and project the atoms produced by the estimators onto said vector. With the same vector, we also compute the corresponding true CDFs of the projected Y given the query point. We then approximately compute the $\mathcal{W}$-distance

between the projected distributions via (19). The resulting histograms are shown in Figure 6, which suggests that LipNet performs best. The rationale for employing this projection approach, rather than directly computing the $\mathcal{W}$-distance between discrete and continuous distributions over $\mathbb{R}^3$, is due to the higher cost and lower accuracy of the latter approach (see also the discussion in Section 5.2). While this projection approach provides a cost-effective alternative for performance evaluation, it may not fully capture the differences between the estimations and ground truth.

Lastly, to demonstrate how atoms, in the neural estimator, move as x varies, Figure 7 shows the projected trajectories along a randomly selected straight line through the origin. The movement of atoms in LipNet is smooth, consistent with previous observations. Interestingly, the movement of atoms in StdNet isn't excessively oscillatory either, although its continuity is slightly rougher compared to LipNet. The reader may execute the `Jupyter` notebook on our github repisitory `https://github.com/zcheng-a/LCD_kNN` to explore the projected conditional CDFs and atoms' trajectories for different x values.

4. Proofs

4.1 Auxiliary notations and lemmas

In this section, we will introduce a few technical results that will be used in the subsequent proofs.

We first define

$$R(m) := \sup_{x\in\mathbb{X}} \int_{\mathbb{Y}^m} \mathcal{W}\left(P_x, \frac{1}{m}\sum_{\ell=1}^m \delta_{y_\ell}\right) \bigotimes_{\ell=1}^m P_x(\mathrm{d}y_\ell), \quad m \in \mathbb{N}. \tag{20}$$

We stipulate that $R(0) = 1$. By Fournier and Guillin (2015), we have

$$R(m) \le \widehat{C} \times \begin{cases} m^{-\frac{1}{2}}, & d_{\mathbb{Y}} = 1, \\ m^{-\frac{1}{2}}\ln(m), & d_{\mathbb{Y}} = 2, \\ m^{-\frac{1}{d_{\mathbb{Y}}}}, & d_{\mathbb{Y}} \ge 3, \end{cases} \tag{21}$$

for some constant $\widehat{C} > 0$ depending only on $d_{\mathbb{Y}}$. For comprehension, we also point to Kloeckner (2020); Fournier (2023) for results that are potentially useful in analyzing explicit constant, though it is out of the scope of this paper.

The lemma below pertains to the so-called approximation error, which arises when treating data points Y_j with X_j around an query point as though they are generated from the conditional distribution at the query point.

Lemma 16 *Under Assumption 2, for any integer $J \ge 1$ and $x, x_1, \ldots, x_J \in \mathbb{X}^{J+1}$, we have*

$$\left| \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^J P_{x_j}(\mathrm{d}y_j) - \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^J P_x(\mathrm{d}y_j) \right| \le \frac{L}{J}\sum_{j=1}^J \|x_j - x\|_\infty.$$

Proof For $x, x_1, \ldots, x_J \in \mathbb{X}^{J+1}$, note that

$$\begin{aligned}
&\left| \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^J P_{x_j}(\mathrm{d}y_j) - \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{k=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^J P_x(\mathrm{d}y_j) \right| \\
&\quad \le \sum_{\ell=1}^J \left| \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^{\ell-1} P_x(\mathrm{d}y_j) \otimes \bigotimes_{j=\ell}^J P_{x_j}(\mathrm{d}y_j) \right. \\
&\qquad\qquad \left. - \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^{\ell} P_x(\mathrm{d}y_j) \otimes \bigotimes_{j=\ell+1}^J P_{x_j}(\mathrm{d}y_j) \right|,
\end{aligned}$$

where for the sake of neatness, at $\ell = 1, J$, we set

$$\bigotimes_{j=1}^0 P_x(\mathrm{d}y_j) \otimes \bigotimes_{j=1}^J P_{x_j}(\mathrm{d}y_j) = \bigotimes_{j=1}^J P_{x_j}(\mathrm{d}y_j) \quad \text{and} \quad \bigotimes_{j=1}^J P_x(\mathrm{d}y_j) \otimes \bigotimes_{j=J+1}^J P_{x_j}(\mathrm{d}y_j) = \bigotimes_{j=1}^J P_{x_j}(\mathrm{d}y_j).$$

Regarding the ℓ-th summand, invoking Fubini-Toneli theorem to integrate y_ℓ first then combining the integrals on outer layers using linearity, we obtain

$$\begin{aligned}
&\left| \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^{\ell-1} P_x(\mathrm{d}y_j) \otimes \bigotimes_{j=\ell}^J P_{x_j}(\mathrm{d}y_j) \right. \\
&\qquad \left. - \int_{\mathbb{Y}^J} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) \bigotimes_{j=1}^{\ell} P_j(\mathrm{d}y_j) \otimes \bigotimes_{j=\ell+1}^J P_{x_j}(\mathrm{d}y_j) \right| \\
&= \left| \int_{\mathbb{Y}^{J-1}} \int_{\mathbb{Y}} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) (P_x - P_{x_\ell})(\mathrm{d}y_\ell) \bigotimes_{j=1}^{\ell-1} \mathrm{d}P_x(y_j) \otimes \bigotimes_{j=\ell+1}^J P_{x_j}(\mathrm{d}y_j) \right| \\
&\le \sup_{(y_j)_{j\neq\ell} \in \mathbb{Y}^{J-1}} \left| \int_{\mathbb{Y}} \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right) (P_{x_\ell} - P_x)(\mathrm{d}y_\ell) \right| \\
&\le \frac{1}{J}\mathcal{W}(P_{x_\ell}, P_x) \le \frac{L}{J}\|x_\ell - x\|_\infty,
\end{aligned}$$

where in the second last inequality we have invoked Kantorovich-Rubinstein duality (cf. (Villani, 2008, Particular case 5.16)) and the fact that, for all $(y_j)_{j\neq\ell} \in \mathbb{Y}^{J-1}$, the map $y_\ell \mapsto \mathcal{W}\left(\frac{1}{J}\sum_{j=1}^J \delta_{y_j}, P_x\right)$ is $\frac{1}{J}$-Lipschitz, and where in the last equality, we have used Assumption 2. ■

We will be using the lemma below, which regards the stochastic dominance between two binomial random variables.

Lemma 17 *Let $n \in \mathbb{N}$ and $0 \le p < p' \le 1$. Then,* Binomial(n, p') *stochastically dominates* Binomial(n, p).

Proof Let $U_1, \ldots, U_n \overset{\text{i.i.d.}}{\sim} \text{Uniform}[0,1]$ and define

$$H := \sum_{i=1}^n \mathbb{1}_{[0,p]}(U_i), \quad H' := \sum_{i=1}^n \mathbb{1}_{[0,p']}(U_i).$$

Clearly, $H \sim \text{Binomial}(n,p)$ and $H' \sim \text{Binomial}(n,p')$. Moreover, we have $H \leq H'$, and thus $\mathbb{P}(H > r) \leq \mathbb{P}(H' > r)$, which completes the proof. ■

4.2 Proof of Theorem 7

The proof of Theorem 7 relies the technical lemma below that we state and prove now.

Lemma 18 *Let $p \in [0,1]$ a real number, and let $M \geq 1$ and $d \geq 1$ two integers. We then have*

$$\sum_{m=1}^{M} \binom{M}{m} p^m (1-p)^{M-m} m^{-\frac{1}{d}} \leq ((M+1)p)^{-\frac{1}{d}} + ((M+1)p)^{-1}.$$

Proof We compute

$$\begin{aligned}
\sum_{m=1}^{M} \binom{M}{m} p^m (1-p)^{M-m} m^{-\frac{1}{d}} &= \frac{1}{(M+1)p} \sum_{m=1}^{M} \binom{M+1}{m+1} p^{m+1} (1-p)^{M-m} (m+1) m^{-\frac{1}{d}} \\
&= \frac{1}{(M+1)p} \sum_{m=2}^{M+1} \binom{M+1}{m} p^m (1-p)^{M+1-m} m (m-1)^{-\frac{1}{d}} \\
&= \frac{1}{(M+1)p} \sum_{m=2}^{M+1} \binom{M+1}{m} p^m (1-p)^{M+1-m} (m-1)^{1-\frac{1}{d}} \\
&\quad + \frac{1}{(M+1)p} \sum_{m=2}^{M} \binom{M+1}{m} p^m (1-p)^{M+1-m} (m-1)^{-\frac{1}{d}},
\end{aligned}$$

where we used that $m = m - 1 + 1$ in the last equality. Then, using that $(m-1)^{1-\frac{1}{d}} \leq m^{1-\frac{1}{d}}$ and $(m-1)^{-\frac{1}{d}} \leq 1$ for all $m \geq 2$, we continue to obtain

$$\begin{aligned}
\sum_{m=1}^{M} \binom{M}{m} p^m (1-p)^{M-m} m^{-\frac{1}{d}} &\leq \frac{1}{(M+1)p} \sum_{m=2}^{M+1} \binom{M+1}{m} p^m (1-p)^{M+1-m} m^{1-\frac{1}{d}} \\
&\quad + \frac{1}{(M+1)p} \sum_{m=2}^{M} \binom{M+1}{m} p^m (1-p)^{M+1-m} \\
&\leq \frac{1}{(M+1)p} \sum_{m=0}^{M+1} \binom{M+1}{m} p^m (1-p)^{M+1-m} m^{1-\frac{1}{d}} + \frac{1}{(M+1)p},
\end{aligned}$$

where the second term in the last equality are derived from the binomial formula. Finally, introducing a random variable V with binomial distribution $\mathcal{B}(M+1,p)$, and using Jensen inequality for the concave function $\mathbb{R}^+ \ni x \mapsto x^{1-\frac{1}{d}} \in \mathbb{R}^+$, we obtain

$$\begin{aligned}
\sum_{m=1}^{M} \binom{M}{m} p^m (1-p)^{M-m} m^{-\frac{1}{d}} &\leq \frac{1}{(M+1)p} \mathbb{E}\left[V^{1-\frac{1}{d}}\right] + \frac{1}{(M+1)p} \\
&\leq \frac{((M+1)p)^{1-\frac{1}{d}}}{(M+1)p} + \frac{1}{(M+1)p} = ((M+1)p)^{-\frac{1}{d}} + ((M+1)p)^{-1},
\end{aligned}$$

which conclude the proof. ■

We are now ready to prove Theorem 7.

Proof [Proof of Theorem 7]

For $\nu \in \mathcal{P}(\mathbb{X})$, we obviously have

$$\mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}(P_x, \hat{P}_x^r)\nu(\mathrm{d}x)\right] \leq \sup_{x\in\mathbb{X}} \mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right],$$

we then focus on proving the right hand side inequality in Theorem 7. To this end, we fix $x \in \mathbb{X}$ and, to alleviate the notations, we let $B := \mathcal{B}^r(x)$ as introduced in Definition 5. Let $N_B := \sum_{m=1}^M \mathbb{1}_B(X_m)$. By Definition 5 and Assumption 3 (i), we have

$$\begin{aligned}
\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] &= \mathbb{E}\left[\mathcal{W}(P_x, \hat{\mu}_B^{\mathcal{D}})\right] = \sum_{m=0}^M \mathbb{E}\left[\mathbb{1}_{N_B=m}\mathcal{W}(P_x, \hat{\mu}_B^{\mathcal{D}})\right] \\
&= \sum_{m=1}^M \binom{M}{m} \mathbb{E}\left[\mathbb{1}_{X_1,\dots,X_m\in B}\mathbb{1}_{X_{m+1},\dots,X_M\notin B}\mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{Y_l}, P_x\right)\right] + \mathbb{P}\left[X_1,\dots,X_M \notin B\right]\mathcal{W}(\lambda_{\mathbb{Y}}, P_x) \\
&\leq \sum_{m=1}^M \binom{M}{m} \mathbb{P}\left[X_{m+1},\dots,X_M \notin B\right]\mathbb{E}\left[\mathbb{1}_{X_1,\dots,X_m\in B}\mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{Y_l}, P_x\right)\right] + \xi(B^c)^M R(0) \\
&= \sum_{m=1}^M \binom{M}{m} \mu(B^c)^{M-m} \int_{(B\times\mathbb{Y})^m} \mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{y_l}, P_x\right) \bigotimes_{\ell=1}^m \psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell) + \xi(B^c)^M R(0). \qquad (22)
\end{aligned}$$

To compute the integral terms, observe that, for fixed $m \geq 1$, by definition of $R(m)$ in (20), Lemma 16 and Remark 6,

$$\begin{aligned}
&\int_{(B\times\mathbb{Y})^m} \mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{y_l}, P_x\right) \bigotimes_{\ell=1}^m \psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell) = \int_{B^m}\int_{\mathbb{Y}^m} \mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{y_l}, P_x\right) \bigotimes_{l=1}^m P_{x_l}(\mathrm{d}y_l) \bigotimes_{\ell=1}^m \xi(\mathrm{d}x_\ell) \\
&\leq \int_{B^m} \left(\int_{\mathbb{Y}^m} \mathcal{W}\left(\frac{1}{m}\sum_{l=1}^m \delta_{y_l}, P_x\right) \bigotimes_{\ell=1}^m P_x(\mathrm{d}y_\ell) + \frac{L}{m}\sum_{\ell=1}^m \|x_\ell - x\|_\infty\right) \bigotimes_{\ell=1}^m \xi(\mathrm{d}x_\ell) \\
&\leq \int_{B^m} (R(m) + 2Lr) \bigotimes_{\ell=1}^m \xi(\mathrm{d}x_\ell) = (R(m) + 2Lr)\xi(B)^m.
\end{aligned}$$

This together with (22) implies that, for any $x \in \mathbb{X}$,

$$\begin{aligned}
\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] &\leq \sum_{m=1}^M \binom{M}{m} \xi(B^c)^{M-m}\xi(B)^m(R(m) + 2Lr) + \xi(B^c)^M R(0) \\
&\leq 2Lr + \sum_{m=1}^M \binom{M}{m} \xi(B^c)^{M-m}\xi(B)^m R(m) + \xi(B^c)^M R(0). \qquad (23)
\end{aligned}$$

The remainder of the proof is split into three cases. In order to proceed, we will put together (21), Lemma 18, and (23). Below we only keep track of the rate.

- For $d_{\mathbb{Y}} = 1$, we have

$$\begin{aligned}
\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] &\leq 2Lr + (\xi(B)(M+1))^{-\frac{1}{2}} + (\xi(B)(M+1))^{-1} + (1-\xi(B))^M \\
&\leq 2Lr + \left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-\frac{1}{2}} + \left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-1} + e^{-\underline{c}Mr^{d_{\mathbb{X}}}}.
\end{aligned}$$

Controlling the dominating term(s) by setting $r \sim r^{-\frac{d_{\mathbb{X}}}{2}} M^{-\frac{1}{2}}$, we yield

$$r \sim M^{-\frac{1}{d_{\mathbb{X}}+2}} \quad \text{and} \quad \mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+2}}.$$

- For $d_{\mathbb{Y}} = 2$, we have

$$\begin{aligned}\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] &\leq 2Lr + \ln(M)\left(\xi(B)(M+1)\right)^{-\frac{1}{2}} + \left(\xi(B)(M+1)\right)^{-1} + (1-\xi(B))^M \\ &\leq 2Lr + \ln(M)\left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-\frac{1}{2}} + \left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-1} + e^{-\underline{c}Mr^{d_{\mathbb{X}}}}.\end{aligned}$$

  Since $r \sim \ln(M) r^{-\frac{d_{\mathbb{X}}}{2}} M^{-\frac{1}{2}}$ may not have a closed-form solution, we simply follow the case of $d_{\mathbb{Y}} = 1$ to yield

$$r \sim M^{-\frac{1}{d_{\mathbb{X}}+2}} \quad \text{and} \quad \mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+2}} \ln M.$$

- For $d_{\mathbb{Y}} \geq 3$, we have

$$\begin{aligned}\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] &\leq 2Lr + \left(\xi(B)(M+1)\right)^{-\frac{1}{d_{\mathbb{Y}}}} + \left(\xi(B)(M+1)\right)^{-1} + (1-\xi(B))^M \\ &\leq 2Lr + \left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-\frac{1}{d_{\mathbb{Y}}}} + \left(\underline{c}(2r)^{d_{\mathbb{X}}}(M+1)\right)^{-1} + e^{-\underline{c}Mr^{d_{\mathbb{X}}}}.\end{aligned}$$

  By setting $r \sim r^{-\frac{d_{\mathbb{X}}}{d_{\mathbb{Y}}}} M^{-\frac{1}{d_{\mathbb{Y}}}}$, we yield

$$r \sim M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}} \quad \text{and} \quad \mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}.$$

The proof is complete. ■

4.3 Proof of Theorem 8

Proof [Proof of Theorem 8] We will proceed by using Efron-Stein inequality. Let (X_1', Y_1') be an independent copy of (X_1, Y_1), and define $\mathcal{D}' := \{(X_1', Y_1'), (X_2, Y_2), \ldots, (X_M, Y_M)\}$. In view of Assumption 3 (i), by the triangle inequality of $\mathcal{W}$, it is sufficient to investigate

$$\frac{1}{2} M \, \mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\left(\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right) \mathrm{d}\nu(x)\right)^2\right].$$

Notice that, by definitions (1),

$$\left\{\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)} \neq \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right\} \subseteq \left\{X_1 \in \mathcal{B}^r(x)\right\} \cup \left\{X_1' \in \mathcal{B}^r(x)\right\}.$$

Additionally, by definitions (1) again, on the event that $\left\{\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)} \neq \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right\}$, we have

$$\mathcal{W}\left(\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right) \leq \left(1 + \sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-1}.$$

The above together with the condition that ν is dominated by $\lambda_{\mathbb{X}}$ implies that

$$\begin{aligned}
\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\left(\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right) \nu(\mathrm{d}x)\right)^2\right] &\leq \overline{C}^2 \mathbb{E}\left[\left(\int_{B(X_1,2r)\cup B(X_1',2r)} \mathcal{W}\left(\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right) \lambda_{\mathbb{X}}(\mathrm{d}x)\right)^2\right] \\
&\leq \overline{C}^2 \mathbb{E}\left[\left(\int_{B(X_1,2r)\cup B(X_1',2r)} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-1} \lambda_{\mathbb{X}}(\mathrm{d}x)\right)^2\right] \\
&\leq 4\overline{C}^2 \mathbb{E}\left[\left(\int_{B(X_1,2r)} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-1} \lambda_{\mathbb{X}}(\mathrm{d}x)\right)^2\right] \\
&= 4\overline{C}^2 \mathbb{E}\left[\lambda_{\mathbb{X}}(B(X_1,2r))^2 \left(\int_{B(X_1,2r)} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-1} \frac{\lambda_{\mathbb{X}}(\mathrm{d}x)}{\lambda_{\mathbb{X}}(B(X_1,2r))}\right)^2\right] \\
&\leq 4\overline{C}^2 (4r)^{2d_{\mathbb{X}}} \mathbb{E}\left[\mathbb{E}\left[\int_{B(X_1,2r)} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-2} \frac{\lambda_{\mathbb{X}}(\mathrm{d}x)}{\lambda_{\mathbb{X}}(B(X_1,2r))}\middle| X_1\right]\right], \qquad (24)
\end{aligned}$$

where we have used Jensen's inequality and tower property in the last line. In view of Assumption 3 (i), expanding the inner conditional expectation into an integral with respect to regular conditional distribution (cf. (Bogachev, 2007, Section 10)) then invoking Fubini-Tonelli theorem, we yield

$$\begin{aligned}
&\mathbb{E}\left[\int_{B(X_1,2r)} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell)\right)^{-2} \frac{\lambda_{\mathbb{X}}(\mathrm{d}x)}{\lambda_{\mathbb{X}}(B(X_1,2r))}\middle| X_1\right] \\
&\quad = \int_{B(X_1,2r)} \int_{\mathbb{X}^{M-1}} \left(1+\sum_{\ell=2}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-2} \bigotimes_{\ell=2}^{M} \xi(\mathrm{d}x_\ell) \frac{\lambda_{\mathbb{X}}(\mathrm{d}x)}{\lambda_{\mathbb{X}}(B(X_1,2r))}. \qquad (25)
\end{aligned}$$

For the inner integral in (25), by Assumption 3 (ii), we have

$$\begin{aligned}
&\int_{\mathbb{X}^{M-1}} \left(1+\sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-2} \bigotimes_{\ell=2}^{M} \xi(\mathrm{d}x_\ell) \\
&= \sum_{\ell=0}^{M-1} \binom{M-1}{\ell} \xi\big(\mathcal{B}^r(x)\big)^{\ell} \Big(1-\xi^r\big(\mathcal{B}^r(x)\big)\Big)^{M-1-\ell} (1+\ell)^{-2} \\
&= \frac{1}{M(M+1)\xi\big(\mathcal{B}^r(x)\big)^2} \sum_{\ell=0}^{M-1} \binom{M+1}{\ell+2} \xi\big(\mathcal{B}^r(x)\big)^{\ell+2} \Big(1-\xi\big(\mathcal{B}^r(x)\big)\Big)^{M-1-\ell} \frac{\ell+2}{\ell+1} \\
&\leq \frac{2}{M(M+1)\xi\big(\mathcal{B}^r(x)\big)^2} \sum_{\ell=2}^{M+1} \binom{M+1}{\ell} \xi\big(\mathcal{B}^r(x)\big)^{\ell} \Big(1-\xi\big(\mathcal{B}^r(x)\big)\Big)^{M+1-\ell} \\
&\leq \frac{2}{M(M+1)\xi\big(\mathcal{B}^r(x)\big)^2}.
\end{aligned}$$

This together with (24), (25) and Assumption 3 (ii) implies

$$\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\left(\hat{\mu}^{\mathcal{D}}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{B}^r(x)}\right) \mathrm{d}\nu(x)\right)^2\right] \leq 8 \frac{2^{2d_{\mathbb{X}}} \overline{C}^2}{\underline{c}^2 M(M+1)}.$$

Invoking Efron-Stein inequality, we conclude the proof. ■

4.4 Proof of Theorem 10

In order to prove Theorem 10, we first establish a few technical lemmas. The following lemma is a first step toward finding the average rate of k-nearest neigbhor method.

Lemma 19 *Suppose Assumption 2 and 3. Let R be defined in Section 4.1. Then, for any $x \in \mathbb{X}$, we have*

$$\mathbb{E}\left[\mathcal{W}\left(P_x, \check{P}_x^k\right)\right] \leq R(k) + \frac{L}{k}\sum_{m=1}^{k}\mathbb{E}\left[Z_x^{(m)}\right],$$

where $Z_{(m)}^x, m = 1, \dots, M$ are the order statistics of $(\|X_m - x\|_\infty)_{m=1}^M$ in ascending order.

Proof We fix $x \in \mathbb{X}$ for the rest of the proof. By Assumption 3, we have

$$\begin{aligned}
\mathbb{E}\left[\mathcal{W}\left(P_x, \check{P}_x^k\right)\right] &= M!\,\mathbb{E}\left[\mathbb{1}_{\|X_1-x\|_\infty \leq \|X_2-x\|_\infty \leq \cdots \leq \|X_M-x\|_\infty}\mathcal{W}\left(P_x, \frac{1}{k}\sum_{\ell=1}^{k}\delta_{Y_\ell}\right)\right] \\
&= M!\int_{(\mathbb{X}\times\mathbb{Y})^M}\mathbb{1}_{\|x_1-x\|_\infty \leq \|x_2-x\|_\infty \leq \cdots \leq \|x_M-x\|_\infty}\mathcal{W}\left(P_x, \frac{1}{k}\sum_{\ell=1}^{k}\delta_{y_\ell}\right)\bigotimes_{\ell=1}^{M}\psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell) \\
&= M!\int_{\mathbb{X}^M}\mathbb{1}_{\|x_1-x\|_\infty \leq \|x_2-x\|_\infty \leq \cdots \leq \|x_M-x\|_\infty}\int_{\mathbb{Y}^k}\mathcal{W}\left(P_x, \frac{1}{k}\sum_{\ell=1}^{k}\delta_{y_\ell}\right)\bigotimes_{\ell=1}^{k}P_{x_\ell}(\mathrm{d}y_\ell)\bigotimes_{j=1}^{M}\xi(\mathrm{d}x_\ell).
\end{aligned}$$

In view of Lemma 16, replacing P_{x_ℓ} above with P_x, we have

$$\begin{aligned}
&\mathbb{E}\left[\mathcal{W}\left(P_x, \check{P}_x^k\right)\right] \\
&\leq M!\int_{\mathbb{X}^M}\mathbb{1}_{\|x_1-x\|_\infty \leq \|x_2-x\|_\infty \leq \cdots \leq \|x_M-x\|_\infty}\int_{\mathbb{Y}^k}\mathcal{W}\left(P_x, \frac{1}{k}\sum_{\ell=1}^{k}\delta_{y_\ell}\right)\bigotimes_{\ell=1}^{k}P_x(\mathrm{d}y_\ell)\bigotimes_{j=1}^{M}\xi(\mathrm{d}x_\ell) \\
&\quad + \frac{L}{k}\sum_{\ell=1}^{k}M!\int_{\mathbb{X}^M}\mathbb{1}_{\|x_1-x\|_\infty \leq \|x_2-x\|_\infty \leq \cdots \leq \|x_M-x\|_\infty}d_{\mathbb{X}}(x_\ell, x)\bigotimes_{j=1}^{M}\xi(\mathrm{d}x_\ell) \\
&= \int_{\mathbb{Y}^k}\mathcal{W}\left(\frac{1}{k}\sum_{l=1}^{k}\delta_{y_l}, P_x\right)\bigotimes_{\ell=1}^{k}P_x(\mathrm{d}y_\ell) \\
&\quad + \frac{L}{k}\sum_{\ell=1}^{k}M!\int_{\mathbb{X}^M}\mathbb{1}_{\|x_1-x\|_\infty \leq \|x_2-x\|_\infty \leq \cdots \leq \|x_M-x\|_\infty}d_{\mathbb{X}}(x_\ell, x)\bigotimes_{j=1}^{M}\xi(\mathrm{d}x_\ell).
\end{aligned}$$

In view of R defined above (21) and $Z_{(m)}^x$ defined in the statement of this lemma, we conclude the proof. ■

The next lemma provides an upper bound to $\sum_{m=1}^{k}\mathbb{E}\left[Z_x^{(m)}\right]$ listed in Lemma 19.

Lemma 20 *Let $Z_{(m)}^x$ be defined as in Lemma 19. Under Assumption 3, for any $x \in \mathbb{X}$, we have*

$$\sum_{m=1}^{k}\mathbb{E}\left[Z_{(m)}^x\right] \leq \frac{2}{\underline{c}^{\frac{1}{d_{\mathbb{X}}}}d_{\mathbb{X}}}\frac{M!}{\Gamma(M+\frac{1}{d_{\mathbb{X}}}+1)}\sum_{m=1}^{k}\sum_{j=0}^{m-1}\frac{\Gamma(j+\frac{1}{d_{\mathbb{X}}})}{j!}.$$

Proof For any $x \in \mathbb{X}$, we compute, since $Z^x_{(m)} \in [0,1]$,

$$\mathbb{E}\left[Z^x_{(m)}\right] = \int_0^1 \mathbb{P}\left[Z^x_{(m)} \geq r\right] \mathrm{d}r = \int_0^1 \left(1 - \mathbb{P}\left[Z^x_{(m)} < r\right]\right) \mathrm{d}r,$$

and we observe that $\left\{Z^x_{(m)} < r\right\} = \{N(x,r) \geq m\}$ with $N(x,r) := \sharp\{1 \leq m \leq M \mid \|X_m - x\| < r\}$. We hence have

$$\mathbb{E}\left[Z^x_{(m)}\right] = \int_0^1 \left(1 - \mathbb{P}\left[N(x,r) \geq m\right]\right) \mathrm{d}r.$$

Since $N(x,r) \sim \mathrm{Binomial}(M, \xi(B(x,r)))$ and $\xi(B(x,r)) \geq \underline{c}\lambda_{\mathbb{X}}(B(x,r)) \geq \underline{c}\frac{r^{d_{\mathbb{X}}}}{2^{d_{\mathbb{X}}}}$ by Assumption 3 (ii), we obtain that $\mathbb{P}\left[N(x,r) \geq m\right] \geq \mathbb{P}\left[N'(x,r) \geq m\right]$ with $N'(x,r) \sim \mathrm{Binomial}(M, \underline{c}\frac{r^{d_{\mathbb{X}}}}{2^{d_{\mathbb{X}}}})$ due to Lemma 17. This implies

$$\begin{aligned}
\mathbb{E}\left[Z^x_{(m)}\right] &\leq \int_0^1 \left(1 - \mathbb{P}\left[N'(x,r) \geq m\right]\right) \mathrm{d}r = \int_0^1 \mathbb{P}\left[N'(x,r) < m\right] \mathrm{d}r \\
&= \sum_{j=0}^{m-1} \binom{M}{j} \int_0^1 \left(\underline{c}\frac{r^{d_{\mathbb{X}}}}{2^{d_{\mathbb{X}}}}\right)^j \left(1 - \underline{c}\frac{r^{d_{\mathbb{X}}}}{2^{d_{\mathbb{X}}}}\right)^{M-j} \mathrm{d}r \\
&= \frac{2}{\underline{c}^{\frac{1}{d_{\mathbb{X}}}} d_{\mathbb{X}}} \sum_{j=0}^{m-1} \binom{M}{j} \int_0^{\frac{\underline{c}}{2^{d_{\mathbb{X}}}}} r^{\frac{1}{d_{\mathbb{X}}}+j-1} (1-r)^{M-j} \mathrm{d}r \\
&\leq \frac{2}{\underline{c}^{\frac{1}{d_{\mathbb{X}}}} d_{\mathbb{X}}} \sum_{j=0}^{m-1} \frac{\Gamma(M+1)}{\Gamma(j+1)\Gamma(M-j+1)} \frac{\Gamma(\frac{1}{d_{\mathbb{X}}}+j)\Gamma(M-j+1)}{\Gamma(\frac{1}{d_{\mathbb{X}}}+M+1)} \\
&= \frac{2M!}{\underline{c}^{\frac{1}{d_{\mathbb{X}}}} d_{\mathbb{X}} \Gamma(\frac{1}{d_{\mathbb{X}}}+M+1)} \sum_{j=0}^{m-1} \frac{\Gamma(\frac{1}{d_{\mathbb{X}}}+j)}{j!},
\end{aligned} \tag{26}$$

and the proof is over. ■

We are now in position to prove Theorem 10.

Proof [Proof of Theorem 10] By combining Lemma 19 and Lemma 20, noting that the upper bound is constant in x, we have

$$\sup_{x \in \mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \leq R(k) + \frac{L}{k} \frac{2M!}{\underline{c}^{\frac{1}{d_{\mathbb{X}}}} d_{\mathbb{X}} \Gamma(M + \frac{1}{d_{\mathbb{X}}} + 1)} \sum_{m=1}^{k} \sum_{j=0}^{m-1} \frac{\Gamma(j + \frac{1}{d_{\mathbb{X}}})}{j!}. \tag{27}$$

Below we only investigate the rate of the right hand side of (27) as $M \to \infty$, and do not keep track of the constant. We first analyze the second term in the right hand side of (27). By Gautschi's inequality (Merkle, 2008, (10.6)), we have

$$\frac{\Gamma(j + \frac{1}{d_{\mathbb{X}}})}{j!} = \frac{\Gamma(j + \frac{1}{d_{\mathbb{X}}})}{\Gamma(j+1)} \leq j^{\frac{1}{d_{\mathbb{X}}}-1}, \quad j \in \{0\} \cup \mathbb{N}.$$

Thus,

$$\sum_{m=1}^{k} \sum_{j=0}^{m-1} \frac{\Gamma(j + \frac{1}{d_{\mathbb{X}}})}{j!} \leq \sum_{m=1}^{k} \sum_{j=0}^{m-1} j^{\frac{1}{d_{\mathbb{X}}}-1} \lesssim \sum_{m=1}^{k} m^{\frac{1}{d_{\mathbb{X}}}} \lesssim k^{1+\frac{1}{d_{\mathbb{X}}}}. \tag{28}$$

By Gautschi's inequality (Merkle, 2008, (12.2)) again, we have

$$\frac{M!}{\Gamma(M+\frac{1}{d_{\mathbb{X}}}+1)} = \frac{\Gamma(M+1)}{\Gamma(M+\frac{1}{d_{\mathbb{X}}}+1)} \leq M^{-\frac{1}{d_{\mathbb{X}}}}. \tag{29}$$

The above implies

$$\sup_{x\in\mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \lesssim R(k) + M^{-\frac{1}{d_{\mathbb{X}}}} k^{\frac{1}{d_{\mathbb{X}}}}.$$

We will split the remainder of the proof into three cases.

- For $d_{\mathbb{Y}} = 1$, by letting $k^{-\frac{1}{2}} \sim M^{-\frac{1}{d_{\mathbb{X}}}} k^{\frac{1}{d_{\mathbb{X}}}}$, we yield

$$k \sim M^{\frac{2}{d_{\mathbb{X}}+2}} \quad \text{and} \quad \sup_{x\in\mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+2}}$$

- For $d_{\mathbb{Y}} = 2$, since the explicit solution of $k^{-\frac{1}{2}} \ln k \sim M^{-\frac{1}{d_{\mathbb{X}}}} k^{\frac{1}{d_{\mathbb{X}}}}$ is elusive, we simply follow the configuration derived in the case of $d_{\mathbb{Y}} = 1$ and yield

$$k \sim M^{\frac{2}{d_{\mathbb{X}}+2}} \quad \text{and} \quad \sup_{x\in\mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+2}} \ln M.$$

- For $d_{\mathbb{Y}} \geq 3$, by letting $k^{-\frac{1}{d_{\mathbb{Y}}}} \sim M^{-\frac{1}{d_{\mathbb{X}}}} k^{\frac{1}{d_{\mathbb{X}}}}$, we yield

$$k \sim M^{\frac{d_{\mathbb{Y}}}{d_{\mathbb{X}}+d_{\mathbb{Y}}}} \quad \text{and} \quad \sup_{x\in\mathbb{X}} \mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \lesssim M^{-\frac{1}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}.$$

The proof is complete. ■

4.5 Proof of Theorem 11

Proof [Proof of Theorem 11] We will proceed by using Efron-Stein inequality. Let (X_1', Y_1') be an independent copy of (X_1, Y_1), and define $\mathcal{D}' := \{(X_1', Y_1'), (X_2, Y_2), \ldots, (X_M, Y_M)\}$. In view of Assumption 3 (i), by the triangle inequality of $\mathcal{W}$, it is sufficient to investigate

$$\frac{1}{2} M\, \mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right].$$

Note that for $\mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)$ to be positive, the event $A_x \cup A_x'$ is necessary, where

$$A_x := \left\{X_1 \in \mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)\right\} \quad \text{and} \quad A_x' := \left\{X_1' \in \mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)\right\}.$$

Moreover,

$$\mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big) \leq \frac{1}{k}.$$

It follows that

$$\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}(x)}}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right] \leq \frac{1}{k^2}\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathbb{1}_{A_x \cup A'_x}\nu(\mathrm{d}x)\right)^2\right] \tag{30}$$
$$\leq \frac{1}{k^2}\mathbb{E}\left[\int_{\mathbb{X}} \mathbb{1}_{A_x \cup A'_x}\nu(\mathrm{d}x)\right] \leq \frac{2}{k^2}\int_{\mathbb{X}} \mathbb{P}\left[A_x\right]\nu(\mathrm{d}x).$$

where the second inequality is due to the fact that the integral value always fall into in $[0,1]$, and we have used Fubini-Tonelli theorem and the subadditivity of probability in the third inequality. Regarding $\mathbb{P}\left[A_x\right]$, by the symmetry stemming from Assumption 3 (i) and the random tie-breaking rule in Definition 9, we have

$$\mathbb{P}\left[A_x\right] = \binom{M-1}{k-1}\binom{M}{k}^{-1} = \frac{k}{M}.$$

Consequently,

$$M\,\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}(x)}}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right] \leq \frac{2}{k}.$$

Invoking Efron-Stein inequality, we conclude the proof of (5).

We now assume additionally that $\nu \leq \overline{C}\lambda_{\mathbb{X}}$ to prove the second statement. Following from (30), by using the positivity and subadditivity of indicator functions as well as AM–GM inequality, we have

$$\begin{aligned}
&\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}(x)}}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right] \\
&\leq \frac{4}{k^2}\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathbb{1}_{A_x}\nu(\mathrm{d}x)\right)^2\right] \leq \frac{4\overline{C}^2}{k^2}\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathbb{1}_{A_x}\lambda_{\mathbb{X}}(\mathrm{d}x)\right)^2\right] \\
&\leq \frac{4\overline{C}^2}{k^2}\int_{[0,1]} \mathbb{P}\left[\left(\int_{\mathbb{X}} \mathbb{1}_{A_x}\lambda_{\mathbb{X}}(\mathrm{d}x)\right)^2 > \delta\right]\mathrm{d}\delta,
\end{aligned}$$

where in the second inequality we have used the condition that ν is dominated by $\lambda_{\mathbb{X}}$, and in the last one the alternative expression of expectation for positive random variables. Let $\mathsf{Cube}^{\iota}_{\mathbb{X}}$ be the set of cubes within $\mathbb{X}$ with edge length ι. Since ν is dominated by $\lambda_{\mathbb{X}}$, with probability 1 we have

$$\begin{aligned}
A_x &= \left\{\text{at most } (k-1) \text{ of } X_\ell, \ell = 2,\ldots,M, \text{ falls into } B_x^{\|X_1 - x\|_\infty}\right\}, \\
A'_x &= \left\{\text{at most } (k-1) \text{ of } X_\ell, \ell = 2,\ldots,M, \text{ falls into } B_x^{\|X'_1 - x\|_\infty}\right\}.
\end{aligned}$$

It follows that

$$\left\{\sum_{m=2}^{M} \mathbb{1}_B(X_m) > k, \quad \forall B \in \mathsf{Cube}^{\iota}_{\mathbb{X}} \text{ with } \partial B \ni X_1\right\} \subseteq \left\{\int_{\mathbb{X}} \mathbb{1}_{A_x}\lambda(\mathrm{d}x) \leq (2\iota)^{d_{\mathbb{X}}}\right\}.$$

By combining the above and setting $\delta = (2\iota)^{2d_{\mathbb{X}}}$, we yield

$$\begin{aligned}
&\mathbb{E}\left[\left(\int_{\mathbb{X}} \mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}(x)}}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right] \\
&\leq \frac{4\overline{C}^2}{k^2}\int_{[0,1]} \mathbb{P}\left[\frac{1}{M-1}\sum_{m=2}^{M} \mathbb{1}_B(X_m) \leq \frac{k}{M-1}, \exists B \in \mathsf{Cube}^{\frac{1}{2}\delta^{\frac{1}{d_{\mathbb{X}}}}}_{\mathbb{X}}\right]\mathrm{d}\delta.
\end{aligned} \tag{31}$$

In order to proceed, we state and prove a useful technical lemma using the Rademacher complexity technique (cf. (Wainwright, 2019, Section 4)). Below we let $\mathsf{Cube}_{\mathbb{X}}$ be the set of cubes inside $\mathbb{X}$ with edge lengths within $[0,1]$.

Lemma 21 *Let $X_2,\dots,X_M$ be introduced in Assumption 3 (i). For $\varepsilon \geq 0$,*

$$\mathbb{P}\left[\frac{1}{M-1}\sum_{m=2}^{M}\mathbb{1}_B(X_m) \leq \underline{c}\lambda_{\mathbb{X}}(B) - 8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} - \varepsilon, \quad \exists B \in \mathsf{Cube}_{\mathbb{X}}\right] \leq \exp\left(-\frac{M-1}{2}\varepsilon^2\right).$$

Proof Let $\boldsymbol{x}^M = (x_2^M,\dots,x_M^M) \in \mathbb{X}^{M-1}$. To utilize the machinery of Rademacher complexity, we will upper bound the cardinality of the set $\{\mathbb{1}_B(\boldsymbol{x}^M) : B \in \mathsf{Cube}_{\mathbb{X}}\}$, where $\mathbb{1}_B$ applies entry-wise. More precisely, $\mathbb{1}_B(\boldsymbol{x}^M) = (\mathbb{1}_B(x_2^M),\dots,\mathbb{1}_B(x_M^M))$. To start with, we first note that for $d = 1,\dots,d_{\mathbb{X}}$, the projected $(x_{2,d}^M,\dots,x_{M,d}^M)$ at most separates axis-d into M intervals. Additionally, each element in $\{\mathbb{1}_B(\boldsymbol{x}^M) : B \in \mathsf{Cube}_{\mathbb{X}}\}$ corresponds to selecting two intervals (one for starting and one for ending of the cube) on each axis. Therefore, the cardinality is at most $M^{2d_{\mathbb{X}}}$, i.e., $\mathsf{Cube}_{\mathbb{X}}$ has polynomial discrimination $2d_{\mathbb{X}}$. It follows from (Wainwright, 2019, Lemma 4.14 and Theorem 4.10) that, for any $\varepsilon \geq 0$,

$$\mathbb{P}\left[\sup_{B\in\mathsf{Cube}_{\mathbb{X}}}\left|\frac{1}{M-1}\sum_{m=2}^{M}\mathbb{1}_B(X_m) - \xi(B)\right| \geq 8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} + \varepsilon\right] \leq \exp\left(-\frac{M-1}{2}\varepsilon^2\right).$$

Finally, in view of Assumption 3 (ii), we conclude the proof of Lemma 21. ∎

In view of (31) and Lemma 21, for $\delta \in [0,1]$, we consider $\varepsilon \geq 0$ such that

$$\frac{k}{M-1} = \frac{\underline{c}\delta^{\frac{1}{2}}}{2^{d_{\mathbb{X}}}} - 8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} - \varepsilon.$$

Note that this is feasible only if $\frac{4^{d_{\mathbb{X}}}}{\underline{c}^2}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} + \frac{k}{M-1}\right)^2 \leq 1$.[4] It follows that

$$\begin{aligned}
&\mathbb{P}\left[\frac{1}{M-1}\sum_{m=2}^{M}\mathbb{1}_B(X_m) \leq \frac{k}{M-1}, \forall B \in \mathsf{Cube}_{\mathbb{X}}^{\frac{1}{2}\delta^{\frac{1}{2d_{\mathbb{X}}}}}\right] \\
&\quad\leq \begin{cases} 1, & \delta \in \left[0, \frac{4^{d_{\mathbb{X}}}}{\underline{c}^2}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} + \frac{k}{M-1}\right)^2\right], \\ \exp\left(-\frac{M-1}{2}\left(\frac{\underline{c}\delta^{\frac{1}{2}}}{2^{d_{\mathbb{X}}}} - 8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} - \frac{k}{M-1}\right)^2\right), & \delta \in \left(\frac{2^{d_{\mathbb{X}}}}{\underline{c}}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} + \frac{k}{M-1}\right)^2, 1\right]. \end{cases}
\end{aligned}$$

The above together with (31) implies

$$\begin{aligned}
&\frac{1}{2}M\mathbb{E}\left[\left(\int_{\mathbb{X}}\mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)}, \hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right] \\
&\quad\leq \frac{2\overline{C}^2M}{k^2}\left(\frac{4^{d_{\mathbb{X}}}}{\underline{c}^2}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} + \frac{k}{M-1}\right)^2\right. \\
&\qquad\left. + \int_{\frac{2^{d_{\mathbb{X}}}}{\underline{c}}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}}+\frac{k}{M-1}\right)}^{1}\exp\left(-\frac{M-1}{2}\left(\frac{\underline{c}\eta}{2^{d_{\mathbb{X}}}} - 8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}} - \frac{k}{M-1}\right)^2\right)2\eta\,\mathrm{d}\eta\right),
\end{aligned}$$

4. We do not include this condition in the statement of Theorem 11, as the bound presented remains valid, albeit vacuous, if this condition is not met.

where we have performed a change of variable $\eta = \delta^{\frac{1}{2}}$ in the last line. Relating to exponential and normal density functions, we calculate the integral to obtain

$$\frac{1}{2}M\mathbb{E}\left[\left(\int_{\mathbb{X}}\mathcal{W}\Big(\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}_{\mathbb{X}}}(x)},\hat{\mu}^{\mathcal{D}'}_{\mathcal{N}^{k,\mathcal{D}'_{\mathbb{X}}}}(x)\Big)\nu(\mathrm{d}x)\right)^2\right]$$
$$\leq \frac{2\overline{C}^2M}{k^2}\frac{4^{d_{\mathbb{X}}}}{\underline{c}^2}\left(\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}}+\frac{k}{M-1}\right)^2+\frac{\sqrt{2\pi}}{\sqrt{M-1}}\left(8\sqrt{\frac{2d_{\mathbb{X}}\ln(M)}{M-1}}+\frac{k}{M-1}\right)+\frac{4}{M-1}\right),$$

where we note the right hand side is of $O\left(\left(\frac{\sqrt{\ln(M)}}{k}+\frac{1}{\sqrt{M}}\right)^2+\frac{1}{k}\left(\frac{\sqrt{\ln(M)}}{k}+\frac{1}{\sqrt{M}}\right)+\frac{1}{k^2}\right)$. Invoking Efron-Stein inequality, we conclude the proof. ■

4.6 Proof of Proposition 13

Proof [Proof of Proposition 13] By triangle inequality,

$$\mathbb{E}\left[\int_{\mathbb{X}}\mathcal{W}(P_x,\tilde{P}^{\Theta}_x)\,\mathrm{d}x\right]$$
$$\leq \mathbb{E}\left[\int_{\mathbb{X}}\mathcal{W}(P_x,P^{\Theta}_x)\left(\lambda_{\mathbb{X}}-\frac{1}{N}\sum_{n=1}^N\delta_{\tilde{X}_n}\right)(\mathrm{d}x)\right]$$
$$+\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N\mathcal{W}(P_{\tilde{X}_n},\overline{P}_{\tilde{X}_n})\right]+\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N\mathcal{W}(\overline{P}_{\tilde{X}_n},\tilde{P}^{\Theta}_{\tilde{X}_n})\right].$$

Then, by Assumption 2 and (7),

$$\mathbb{E}\left[\int_{\mathbb{X}}\mathcal{W}(P_x,P^{\Theta}_x)\left(\lambda_{\mathbb{X}}-\frac{1}{N}\sum_{n=1}^N\delta_{\tilde{X}_n}\right)(\mathrm{d}x)\right]\leq\mathbb{E}\left[(L+L^{\Theta})\mathcal{W}\left(\lambda_{\mathbb{X}},\frac{1}{N}\sum_{n=1}^N\delta_{\tilde{X}_n}\right)\right].$$

In view of Assumption 12, we have

$$\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^N\mathcal{W}(P_{\tilde{X}_n},\overline{P}_{\tilde{X}_n})\right]=\frac{1}{N}\sum_{n=1}^N\mathbb{E}\left[\mathbb{E}\left[\mathcal{W}(P_{\tilde{X}_n},\overline{P}_{\tilde{X}_n})\middle|\tilde{X}_n\right]\right]=\mathbb{E}\left[\int_{\mathbb{X}}\mathcal{W}(P_x,\overline{P}_x)\,\mathrm{d}x\right].$$

Combining the above, we prove the first statement.

As for the second statement, consider $Q,Q':\mathbb{X}\to\mathcal{P}(\mathbb{Y})$ that are Lipschitz-continuous with constants L,L'. Suppose that

$$\mathcal{W}(Q_{x^*},Q'_{x^*})=\sup_{x\in\mathbb{X}}\mathcal{W}(Q_x,Q'_x)=\delta$$

for some $\delta>0$ and $x^*\in\mathbb{X}$. This supremum is indeed attainable because $\mathbb{X}$ is compact that $x\mapsto\mathcal{W}(Q_x,Q'_x)$ is continuous. Consequently, by triangle inequality and the Lipschitz-continuity, we have

$$\mathcal{W}(Q_x,Q'_x)\geq\Big(\mathcal{W}(Q_x,Q'_{x^*})-\mathcal{W}(Q'_x,Q'_{x^*})\Big)\vee 0$$
$$\geq\Big(\mathcal{W}(Q_{x^*},Q'_{x^*})-\mathcal{W}(Q_{x^*},Q_x)-\mathcal{W}(Q'_{x^*},Q'_x)\Big)\vee 0$$
$$\geq\Big(\delta-(L+L')\|x-x^*\|_\infty\Big)\vee 0,\quad x\in\mathbb{X}.$$

We may then lower bound $\int_{\mathbb{X}} \mathcal{W}(Q_x, Q'_x)\,\mathrm{d}x$ with the volume of the cone on right hand side above (note the worst case is when $x^* = (0,0)$),

$$\int_{\mathbb{X}} \mathcal{W}(Q_x, Q'_x)\,\mathrm{d}x \geq \int_0^\delta \left(\frac{\delta - z}{L + L'}\right)^{d_{\mathbb{X}}} \mathrm{d}z = \frac{\delta^{d_{\mathbb{X}}+1}}{(d_{\mathbb{X}}+1)(L+L')^{d_{\mathbb{X}}}}.$$

It follows that

$$\sup_{x\in\mathbb{X}} \mathcal{W}(Q_x, Q'_x) \leq (d_{\mathbb{X}}+1)^{\frac{1}{d_{\mathbb{X}}+1}} (L+L')^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+1}} \left(\int_{\mathbb{X}} \mathcal{W}(Q_x, Q'_x)\,\mathrm{d}x\right)^{\frac{1}{d_{\mathbb{X}}+1}},$$

which completes the proof. ■

5. Implementation details and ablation analysis

In this section, we will provide further implementation details and conduct ablation analysis of the components highlighted in Section 3.1.

5.1 Comparing ANNS-RBSP to exact NNS

Algorithm 3 outlines a single slice of RBSP, which divides an array of x's into two arrays of a random ratio along a random axis. Throughout the training, we execute RBSP 5 times during each training epoch, yielding $2^5 = 32$ parts. Within each part, we then select a small batch of 8 query points, locating the k nearest neighbors for each query point within the same part. In Table 1, we compare the execution times of exact NNS and ANNS-RBSP. ANNS-RBSP offers considerable time savings for $M = 10^6$, while exact NNS is more efficient for $M = 10^5$ or fewer.

Algorithm 3 Single slice of random binary space partitioning

Input: data $\mathsf{D}_{\mathbb{X}} = (x_i)_{i=1}^M \subset [0,1]^{d_{\mathbb{X}}}$, arrays of indexes $\mathsf{S}_d, d = 1, \ldots, d_{\mathbb{X}}$ of length M with the j-th entry indicating the position of the j-th smallest value in the d-th dimension of $\mathsf{D}_{\mathbb{X}}$, a boolean array B of length M with the i-th entry indicating whether x_i is involved in the current slicing, a rectangle R that bounds x_i's involved in the current slicing, i.e., R corresponds to B, a parameter $r_{\text{edge}} \in (1, \infty)$ for avoiding thin rectangles, an interval $[\underline{p}, \overline{p}] \in (0,1)$ for random bisecting ratio

Output: two boolean arrays B, B' of length M indicating the bisected data, two bounding rectangles R, R' that correspond to B, B'

Randomly pick a dimension d
if The edge ratio of R exceeds r_{edge} **then**
    Replace d with that corresponds to the longest edge
end if
Rearrange B according to S_d by $\tilde{\mathsf{B}} \leftarrow \mathsf{B}[\mathcal{S}_d]$
Pick out the indexes from S_d involved in ANNS by $\tilde{\mathsf{S}}_d \leftarrow \mathsf{S}_d[\tilde{\mathcal{B}}]$
Generate $p \sim \text{Uniform}([\underline{p}, \overline{p}])$ and round p into $\tilde{p}$ so that $\tilde{p}\,\mathsf{len}(\tilde{\mathsf{S}}_d)$ is an integer
Bisect $\tilde{\mathsf{S}}_d$ in two arrays with length $\tilde{p}\,\mathsf{len}(\tilde{\mathsf{S}}_d)$ and $(1-\tilde{p})\mathsf{len}(\tilde{\mathsf{S}}_d)$, denoted by $\tilde{\mathsf{S}}_d$ and $\tilde{\mathsf{S}}'_d$
Form new bounding rectangles R, R' using $\tilde{\mathsf{S}}_d, \tilde{\mathsf{S}}'_d, \mathsf{D}_{\mathbb{X}}$ and the original R (may enforce some overlap here)
Initialize two boolean arrays B, B' with length M and all entries being False
$\mathsf{B}[\tilde{\mathsf{S}}_d] \leftarrow \mathsf{True}$, $\mathsf{B}'[\tilde{\mathsf{S}}'_d] \leftarrow \mathsf{True}$
return $\mathsf{B}, \mathsf{B}', R, R'$

It's important to note that ANNS-RBSP may introduce additional errors by inaccurately including points that are not within the k nearest neighbors. As elucidated in the proof of Theorem 10, the magnitude of this induced error can be understood by comparing the excessive distance

Table 1 Execution times of two NNS methods.

k=300			
	$M = 10^4$	$M = 10^5$	$M = 10^6$
$d_{\mathbb{X}} = 1$	(**0.1**, 9.2)	(**3.7**, 11.0)	(19.5, **11.2**)
$d_{\mathbb{X}} = 3$	(**0.2**, 9.8)	(**4.2**, 11.2)	(24.6, **11.3**)
$d_{\mathbb{X}} = 10$	(**0.4**, 9.9)	(**5.9**, 11.4)	(52.4, **11.6**)
k=1000			
	$M = 10^4$	$M = 10^5$	$M = 10^6$
$d_{\mathbb{X}} = 1$	(**0.1**, 7.2)	(**3.8**, 10.9)	(19.6, **11.1**)
$d_{\mathbb{X}} = 3$	(**0.2**, 7.2)	(**4.2**, 11.2)	(24.7, **11.4**)
$d_{\mathbb{X}} = 10$	(**0.4**, 7.4)	(**5.9**, 11.4)	(52.8, **11.5**)

This table compares the execution times for 500 runs of exact NNS versus ANNS-RBSP, both utilizing parallel computing, facilitated by PyTorch, with an NVIDIA L40 GPU. Each iteration (approximately) finds the 300 nearest neighbors from M samples for all of 256 randomly generated query points. The values within each parenthesis denote the seconds consumed by both methods, with the first number corresponding to exact NNS. For faster processing, exact NNS employs a 3D tensor. ANNS-RBSP regenerates a new partition each run. The table does not include the time required to sort the data along all dimensions, which takes about 0.2 seconds in the worst case and is not repeatedly executed.

incurred to that of exact NNS. For simplicity, we investigate the difference below

$$\Delta := \frac{1}{N_{\text{batch}}} \sum_{i=1}^{N_{\text{batch}}} \left(\frac{1}{k} \sum_{j=1}^{k} \left\| \check{X}'_{ij} - \tilde{X}_i \right\|_1 - \frac{1}{k} \sum_{j=1}^{k} \left\| \check{X}_{ij} - \tilde{X}_i \right\|_1 \right),$$

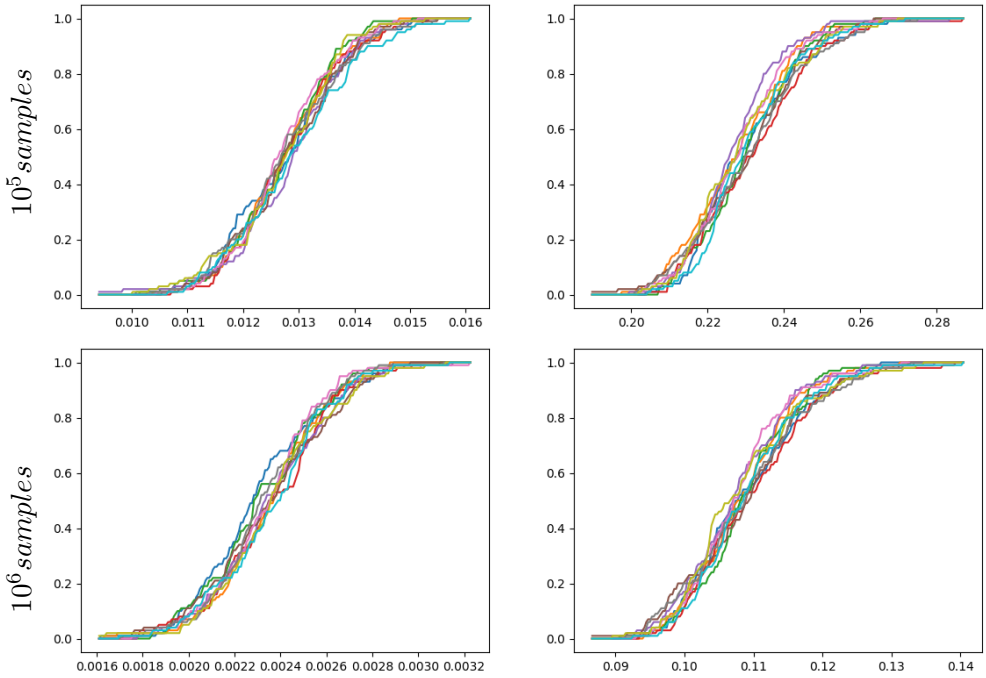
where $\tilde{X}_i$'s are query points, and $\check{X}_{ij}, \check{X}'_{ij}$ are the k-nearest-neighbor identified by exact NNS and ANNS-RBSP, respectively. In our experiments, we evaluated scenarios with $d_{\mathbb{X}} = 3, 10$ and $k = 300$. Regarding the data, we generated $M = 10^4, 10^5, 10^6$ samples from Uniform$([0,1]^{d_{\mathbb{X}}})$. Once the data set is generated, we fixed the data and conducted 100 simulations of Δ, each with $N_{\text{batch}} = 256$ query points. This process was repeated 10 times, each with a separately generated data. The results are illustrated in Figure 8. It is expected that Δ will approach 0 as the sample size M tends to infinity. The convergence rate is likely influenced by factors such as $d_{\mathbb{X}}$, k, and N_{batch}. Further analysis of the convergence of ANNS-RBSP will be conducted in future studies.

5.2 An implementation of the Sinkhorn algorithm

In this section, we will detail our implementation of the Sinkhorn algorithm and highlight a few novel treatments that seem to enhance the training of the neural estimator. While the mechanisms are not yet fully understood, they constitute important improvement in the accuracy of the neural estimator.

Let us first recall the iterative procedure involved in the Sinkhorn algorithm. We follow the setup in Section 3.1.2. In particular, the row indexes of the cost matrix stand for atoms in the empirical measures, while the column indexes stand for atoms produced by the neural estimator. We set $N_{\text{atom}} = k$ and let $\mathsf{u}^{(0)}, \mathsf{v}^{(0)}$ be column vectors of size k with all entries being k^{-1}. We will suppress the dependence on y from the notation. Upon setting

$$\mathsf{K}^{\epsilon} := \exp\left(-\frac{\mathsf{C}}{\epsilon}\right)$$



(a) $d_{\mathbb{X}} = 3$ ***(b)*** $d_{\mathbb{X}} = 10$

Figure 8: *Empirical CDFs of* Δ.

We compare the empirical CDFs of Δ. Each line corresponding to a independently generated set of data. Each plot includes 10 empirical CDFs. Note the difference in the x axis scale.

with entry-wise exponential, the Sinkhorn algorithm performs repeatedly

$$\mathsf{u}^{(\ell+1)} = \frac{\mathsf{u}^{(0)}}{\mathsf{K}^\epsilon \mathsf{v}^{(\ell)}} \quad \text{and} \quad \mathsf{v}^{(\ell+1)} = \frac{\mathsf{v}^{(0)}}{(\mathsf{K}^\epsilon)^\top \mathsf{u}^{(\ell+1)}}, \tag{32}$$

where the division is also calculated entry-wise. After a certain number of iterations, denoted as N_{iter}, we obtain an approximate optimal transport plan for problem (15):

$$\mathsf{T}^\epsilon = \operatorname{diag}(\mathsf{u}^{(N_{\text{iter}})}) \mathsf{K}^\epsilon \operatorname{diag}(\mathsf{v}^{(N_{\text{iter}})}).$$

Let us set $\epsilon = 1$ momentarily. Note that if the entries of C are excessively large, K effectively becomes a zero matrix, which impedes the computations in (32). This issue may occur at the initiation of the neural estimator or during training, possibly due to the use of stochastic gradient descent. To tackle this issue, we employ a rule-of-thump normalization that

$$\tilde{\mathsf{K}}^\epsilon := \exp\left(-\frac{\mathsf{C}}{\tilde{c}\epsilon}\right) \quad \text{with} \quad \tilde{c} := \min_i \max_j \mathsf{C}_{ij}, \tag{33}$$

and use $\tilde{\mathsf{K}}^\epsilon$ instead of K^ϵ in (32). Regarding the selection of ϵ and the number of iterations, we currently lack a method for adaptively determining these values. Instead, we adjust them manually based on training episodes. This manual adjustment works well for all models discussed in this paper. For more information, please see Appendix C.

As alluded in Section 3.1.2, we enforce sparsity on the transport plan to improve the performance of the neural estimator. Let $\tilde{\mathsf{T}}^\epsilon$ be the output of the Sinkhorn algorithm. We construct $\hat{\mathsf{T}}^\epsilon$ and $\check{\mathsf{T}}^\epsilon$ by setting the row-wise and column-wise maximum of $\tilde{\mathsf{T}}^\epsilon$ to k^{-1}, respectively, and setting the remaining entries to 0. We then use

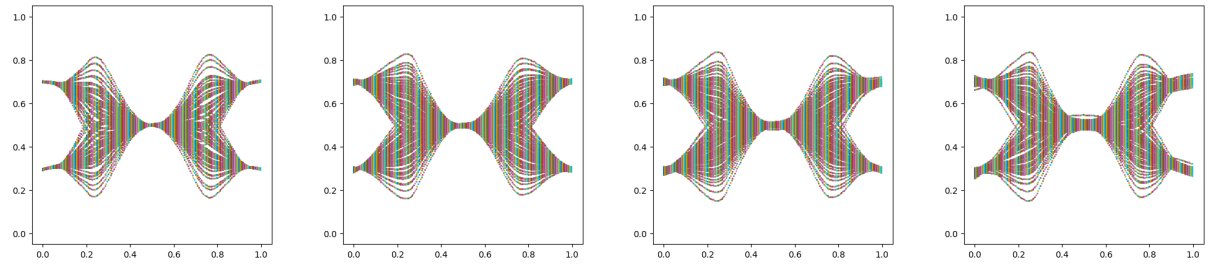
$$\overline{\mathsf{T}}^\epsilon = \gamma \hat{\mathsf{T}}^\epsilon + (1-\gamma)\check{\mathsf{T}}^\epsilon, \tag{34}$$

where $\gamma \in [0,1]$ is a hyper-parameter, in gradient descent (14). We observe that $\hat{\mathsf{T}}^\epsilon$ relates each atom in the empirical measure to a single corresponding atom from the neural estimator, and $\check{\mathsf{T}}^\epsilon$ does the same in reverse. The optimal choice of γ remains an open question, though we have set $\gamma = 0.5$ in all three models.

Next, we explore the impact of enforcing sparsity and varying the choices of γ. Figure 9 compares the performance in Model 1 under different sparsity parameters. When no sparsity is enforced, the neural estimator tend to handles singularities more adeptly, but may overlooks points located on the periphery of the empirical joint distribution, potentially resulting in overly concentrated atoms from the neural estimator (see around $x = 0.1, 0.9$). Compare Figure 4 and 10 for the extra error due to the lack of enforced sparsity. This phenomenon is more noticeable in Model 3. We refer to panel (2,3) of Figure 18 in Appendix B for an example. Moreover, Figure 19, which is obtained without enforced sparsity, indicates a downgrade in accuracy when compared to Figure 6.

For completeness, we present in Table 2 the accuracy of LipNet across various ϵ values, both with and without enforced sparsity. As demonstrated in the table, the improvement resulting from enforced sparsity is evident. It is worth noting that smaller values of ϵ generally require more iterations to achieve convergence (see, e.g., (Peyré and Cuturi, 2019, Section 4.2)). Additionally, other hyper-parameters may also significantly influence the training outcomes. Further exploration of these related issues will be conducted in future studies.

Finally, it is not recommended to use $\overline{\mathsf{T}}^\epsilon$ at the early stages of training, as our empirical experiments suggest this could deteriorates performance. In training, we start by not enforcing sparsity and then begin to enforce it in later episodes. We refer to Appendix C for further details of the training configuration.



(a) *No sparsity enforced* ***(b)*** $\gamma = 0$ ***(c)*** $\gamma = 0.5$ ***(d)*** $\gamma = 1$

Figure 9: *LipNet under Model 1 with different sparsity enforcement.*

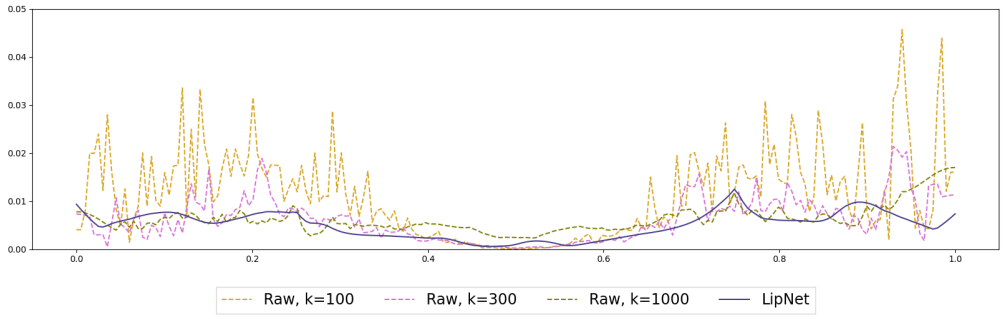


Figure 10: *Errors at different x's of various estimators under Model 1, LipNet is trained without enforced sparsity.*

We compute the $\mathcal{W}$-distance between estimators and the true conditional distribution at different x's. The setting is similar to Figure 4, but LipNet is trained without enforcing sparsity on the transport plan. The errors of LipNet at around $x = 0.1, 0.9$ are slightly higher than those in Figure 4.

Table 2 Projected Wasserstein error under different Sinkhorn parameters.

LipNet without enforced sparsity					
ϵ^{-1}	mean	25%-quantile	median	75%-quantile	number of NaN steps
10	5.18	2.18	3.99	7.34	0
30	1.43	<u>0.83</u>	1.21	1.78	0
100	1.50	0.96	1.31	1.73	1217
300	9.15	4.97	8.62	12.30	9500

LipNet with enforced sparsity					
ϵ^{-1}	mean	25%-quantile	median	75%-quantile	number of NaN steps
10	1.42	0.91	<u>1.20</u>	<u>1.60</u>	0
30	**1.24**	**0.79**	**1.07**	**1.42**	0
100	1.52	0.97	1.32	1.76	0
300	5.00	4.96	5.01	5.05	0

Raw estimator				
k	mean	25%-quantile	median	75%-quantile
300	2.69	1.62	2.22	3.23
1000	1.57	0.95	1.30	1.91
3000	<u>1.41</u>	0.88	1.21	1.70
10000	2.38	1.45	2.02	2.97

This table displays the evaluation error of LipNet across various values of ϵ, both with and without enforced sparsity. For comparison, the error of the raw k-nearest-neighbor estimator is also included. The mean and quantiles are reported at a scale of 10^{-2}. Boldfaced numbers indicate the best results, while underlined numbers denote the runner-up. Note that ϵ is only applied after the first 500 training episodes. For additional details on the configuration, please refer to Table 4. The evaluation is performed by computing the projected Wasserstein distance, as outlined in Section 3.2.

5.3 More on LipNet

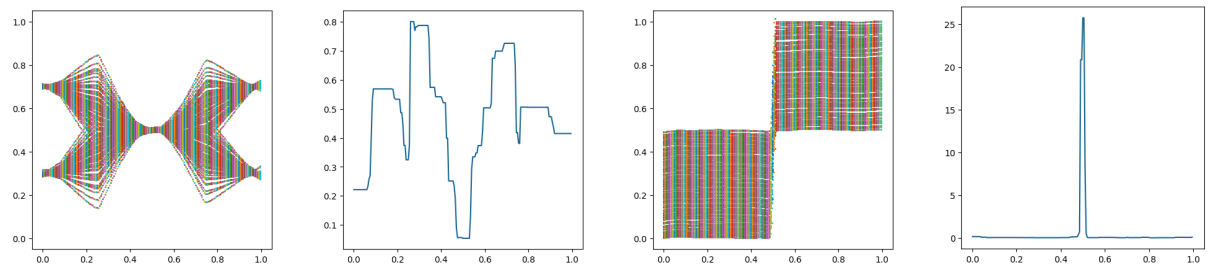
We will investigates the impact of various hyper-parameters on the performance of LipNet. The LipNets presented in this section are trained with the same hyper-parameters as in Section 3.2 (see also Appendix C), expect for those specified otherwise.

5.3.1 Activation function

Switching the activation function from ELU to Rectified Linear Unit (ReLU) appears to retain the adaptive continuity property. In Figure 11, we illustrate the joint distribution and the average absolute derivatives of all atoms of LipNet with ReLU activation. The outcomes are on par with those achieved using ELU activation as shown in Figure 2.

5.3.2 Value of L in (18)

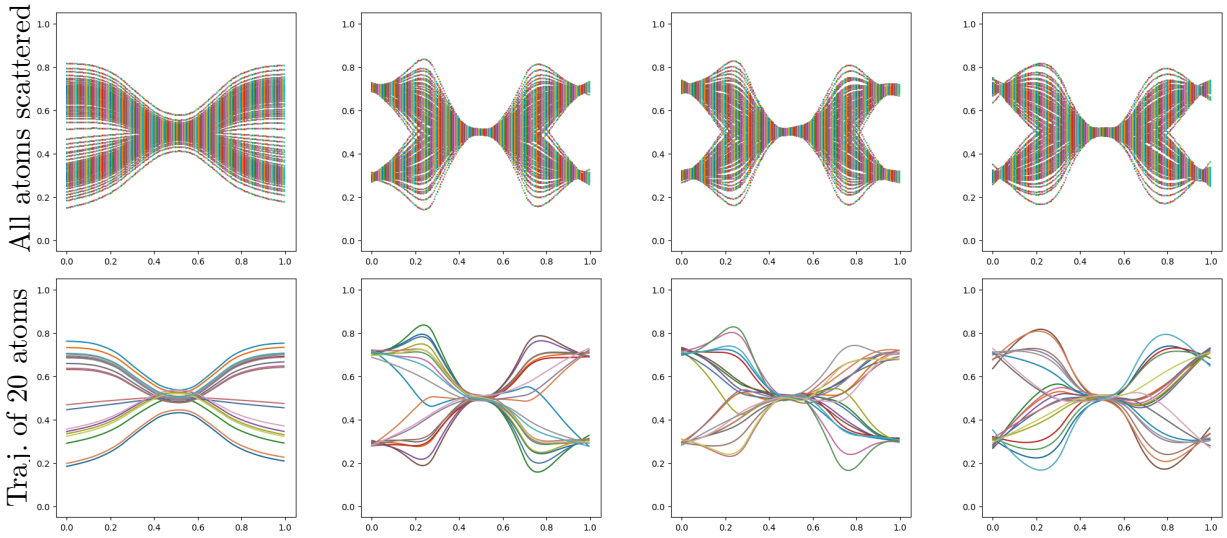
Note that the LipNets discussed in Section 3.2 were trained with $L = 0.1$. If the normalizing constants in LipNet are exactly computed, L reflects the Lipschitz constant of LipNet, upto the discrepancy in the choice of norms in different layers. The effect of L in our implementation, however, is rather obscure. Figure 12 showcases the performance of LipNets across various L values in Model 1. The comparison in Model 2 is presented in Figure 20 in Appendix B. The best choice of L appears to depend on the ground truth model. For Model 3, we compared the performance of $L = 0.1$ and $L = 1$ and observed no significant differences. Generally, we prefer a smaller L;



(a) *Model 1, all atoms* **(b)** *Model 2, ave. abs. der.* **(c)** *Model 2, all atoms* **(d)** *Model 2, ave. abs. der.*

Figure 11: *LipNet under Model 1 with ReLU activation.*

however, smaller values of L tend to exhibit greater sensitivity to other training parameters. For instance, in Model 3, with $L = 0.1$, starting enforcing sparsity too soon leads to significantly poorer performance, while the impact on the outcomes for $L = 1$ is much less noticeable.



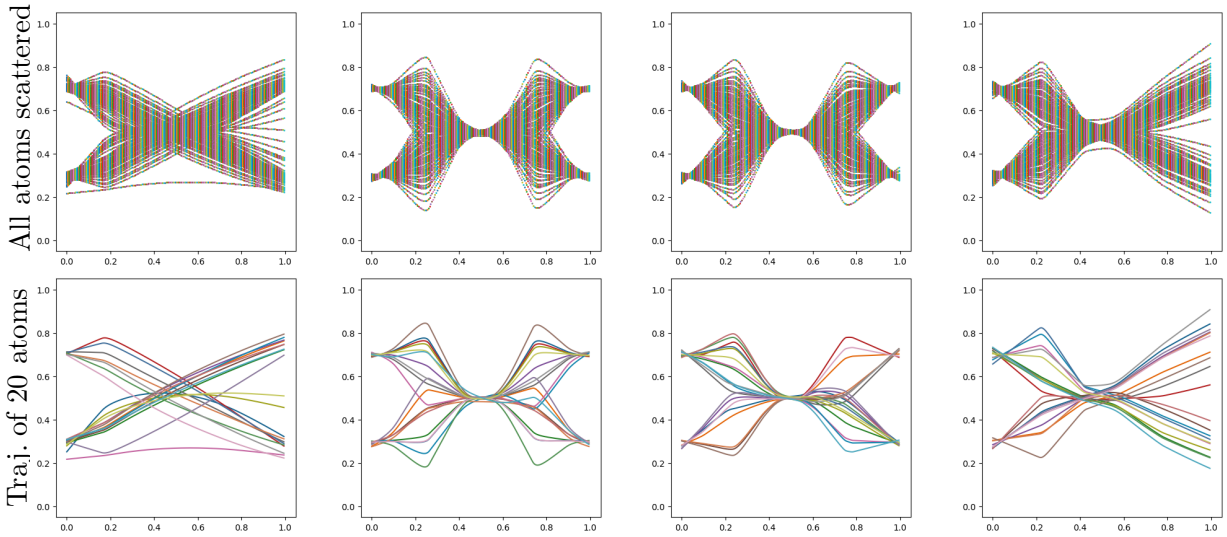
(a) $L = 0.01$ **(b)** $L = 0.03$ **(c)** $L = 1$ **(d)** $L = 3$

Figure 12: *LipNet under Model 1 with various L's.*

5.3.3 Momentum τ in Algorithm 2

In our training of LipNet, we use $\tau = 10^{-3}$. Figure 13 demonstrates the impact of various τ values on neural estimator's performance in Model 1. It is clear that the performance declines with a τ that is too large. While we initially speculated that a smaller τ might cause atoms to exhibit more erratic movements as x changes, observations contradict this hypothesis. We now believe that a suitable τ value helps prevent neurons from stagnating in the plateau region of the ELU activation function. This is supported by the outcomes observed with $\tau = 10^{-6}$, where atom movements are overly simplistic. Additional comparisons in Model 2 are presented in Figure 21.

Despite considering as a potential improvement the inclusion of batch normalization in the convex potential layer (16), right after the affine transformation, along with a corresponding offset in the position of $\|\mathsf{W}\|_2$, our experiments with both ELU and ReLU activations, using the default



(a) $\tau = 10^{-1}$ ***(b)*** $\tau = 10^{-2}$ ***(c)*** $\tau = 10^{-5}$ ***(d)*** $\tau = 10^{-6}$

Figure 13: *LipNet under Model 1 with various τ's.*

batch normalization momentum of 0.1, resulted in reduced performance. Lowering said batch normalization momentum often leads to a NaN network.

6. Weakness and potential improvement

In this section, we provide some discussion on the weakness and possible improvement of our implementation in Section 3.

Extra correction. In more realistic scenarios, the true conditional distribution is often unknown or intractable. In such cases, it is unclear whether a neural estimator offers extra correction over raw estimators. A potential solution to this issue is to train StdNet and LipNet simultaneously. If StdNet and LipNet align more closely with each other than with the raw estimator involved in their training, it is possible that the neural estimators are providing extra corrections.

Hyper-parameters for Sinkhorn algorithm. Our implementation of the Sinkhorn algorithm involves several hyper-parameters: (i) k in Definition 9; (ii) N_{atom} in (10); (iii) ϵ in (33); (iv) γ in (34); and (v) additional hyper-parameters listed in Section C. The impact of these hyper-parameters is not yet fully understood. Additionally, an adaptive ϵ that balances the accuracy and stability of the Sinkhorn iteration is desirable. Furthermore, as illustrated in Section 5.2, enforcing sparsity on the transport plan generally yields better approximations at x where the conditional distribution is more diffusive, but may performs worse where the conditional distribution exhibits atoms. This observation motivates further investigation into a sparsity policy that adjusts according to the indications from the raw estimator.

Adaptive continuity. The impact of hyper-parameters in LipNet also warrants further investigation. In addition, despite the results presented in this study, more evidence is needed to understand how LipNet and its variations perform under various conditions.

Scalability. While the implementation produces satisfactory results when M and k are relatively small (recall that we set $N_{\text{atom}} = k$), our further experiments indicate a scalability bottleneck. For example, in Model 1, significantly increasing M and k does not necessarily improve the performance of neural estimators in a comparable manner. For completeness, we provide a report of the computational time consumption under various settings in Section C. To address this issue, we

could experiment with varying the ratios between N_{atoms} and k, rather than setting them equal, in hopes of reducing the strain on the Sinkhorn algorithm. We note that varying the ratio between N_{atoms} and k requires adjusting the enforced sparsity accordingly. Another issue relates to the dimensions of $\mathbb{X}$ and $\mathbb{Y}$. In view of the curse of dimensionality in Theorem 10, our method is inherently suited for low-dimensional settings. Fortunately, in many practical scenarios, the data exhibits low-dimensional structures, such as: (i) the sampling distribution of X concentrating on a low-dimensional manifold; and (ii) the mapping $x \mapsto P_x$ exhibiting low-dimensional dependence. For (i), we might resort to dimension reduction techniques, although an extension of the results in Section 2 has yet to be established. For (ii), a data-driven method that effectively leverages the low-dimensional dependence is of significant interest.

Conditional generative models. Utilizing a conditional generative model could potentially lead to further improvements. One advantage of conditional generative models is the ease of incorporating various training objectives. For instance, it can easily adapt to the training objectives in (6) to accommodate multiple different hyper-parameters simultaneously. We may also incorporate the joint empirical measure in the training process. This flexibility also allows for the integration of specific tail conditions as needed.

Lastly, we would like to point out an issue observed in our preliminary experiments when utilizing a naïve conditional generative model: it may assign excessive probability mass to the blank region between two distinct clusters (for example, in Model 1 around $(x, y) = (0.1, 0.5)$). This possibly stems from the inherent continuity of neural networks. One possible solution is to consider using a mixture of multiple conditional generative models.

Appendix A. Numerical finite-sample bounds

In this section, we present numerical computations of finite-sample bounds for the r-box and k-nearest-neighbor estimators in the case where $d_{\mathbb{X}} = d_{\mathbb{Y}} = 3$ and $\overline{C} = \underline{c} = 1$, i.e., both the sampling distribution of X and ν are uniform. Through this analysis, we also aim to explore the impact of the Lipschitz constant L and the hyperparameters r and k, as introduced in Sections 2.2 and 2.3.

To begin, we note that an explicit expression for $R(m)$, as defined in (21), can be found in (Kloeckner, 2020, Theorem 2.1). Additionally, we introduce a parameter σ to represent the level of uncertainty in P_x. This uncertainty parameter is motivated by the observation that $R(m)$ is derived as a bound across all probability measures on $[0,1]^{d_{\mathbb{Y}}}$. In scenarios where, for each $x \in \mathbb{X}$, P_x is supported on an cube (possibly dependent on x) of diameter $\sigma^{\frac{1}{d_{\mathbb{Y}}}} \in (0,1)$, we can replace $R(m)$ in (23) and (27) with $\sigma R(m)$. A similar approach applies to other cases, such as small moment conditions. The precise definition of the uncertainty parameter σ and its data-driven estimation will be explored in future work. Here, we use σ to investigate how the magnitude of uncertainty affects the bounds and the corresponding optimal configurations.

For the r-box estimator, we utilize (23). In the current setting, $\xi(B)$ in (23) simplifies to $(2r)^{d_{\mathbb{X}}}$. Consequently, the bound becomes

$$\mathbb{E}\left[\mathcal{W}(P_x, \hat{P}_x^r)\right] \leq 2Lr + \sigma \sum_{m=1}^{M} \binom{M}{m} \left(1 - (2r)^{d_{\mathbb{X}}}\right)^{M-m} (2r)^{d_{\mathbb{X}} m} R(m) + \sigma \left(1 - (2r)^{d_{\mathbb{X}}}\right)^{M} R(0).$$

The second term on the right-hand side can be approximated using Monte Carlo sampling from a binomial distribution with $n = M$ and $p = (2r)^{d_{\mathbb{X}}}$. The Monte Carlo approximation is performed with $1,000$ samples.

For the k-nearest-neighbor estimator, we combine (27) with (28) and (29) to obtain the following upper bound

$$\mathbb{E}\left[\mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{N}^k(x)}\right)\right] \leq \sigma R(k) + \frac{2L}{k} M^{-\frac{1}{d_{\mathbb{X}}}} \sum_{m=1}^{k} \sum_{j=0}^{m-1} j^{\frac{1}{d_{\mathbb{X}}}-1}.$$

Table 3 below reports the optimal values and configurations of r and k based on these bounds, with different L and σ. For visualization, these bounds are also plotted in Figure 14.

Table 3 Finite sample bounds with $d_{\mathbb{X}} = d_{\mathbb{Y}} = 3$, $\overline{C} = \underline{c} = 1$, $M = 10^6$.

$\sigma = 0.1$				
	r-box		k-NN	
L	min. bound (10^{-2})	r^*	min. bound (10^{-2})	k^*
0.1	3.79	0.095	4.49	3548
0.3	6.57	0.055	7.57	707
1	11.00	0.005	11.90	10

$\sigma = 1$				
	r-box		k-NN	
L	min. bound (10^{-2})	r^*	min. bound (10^{-2})	k^*
0.1	12.00	0.3	14.53	112201
0.3	20.78	0.175	24.96	22387
1	37.94	0.095	44.85	3548

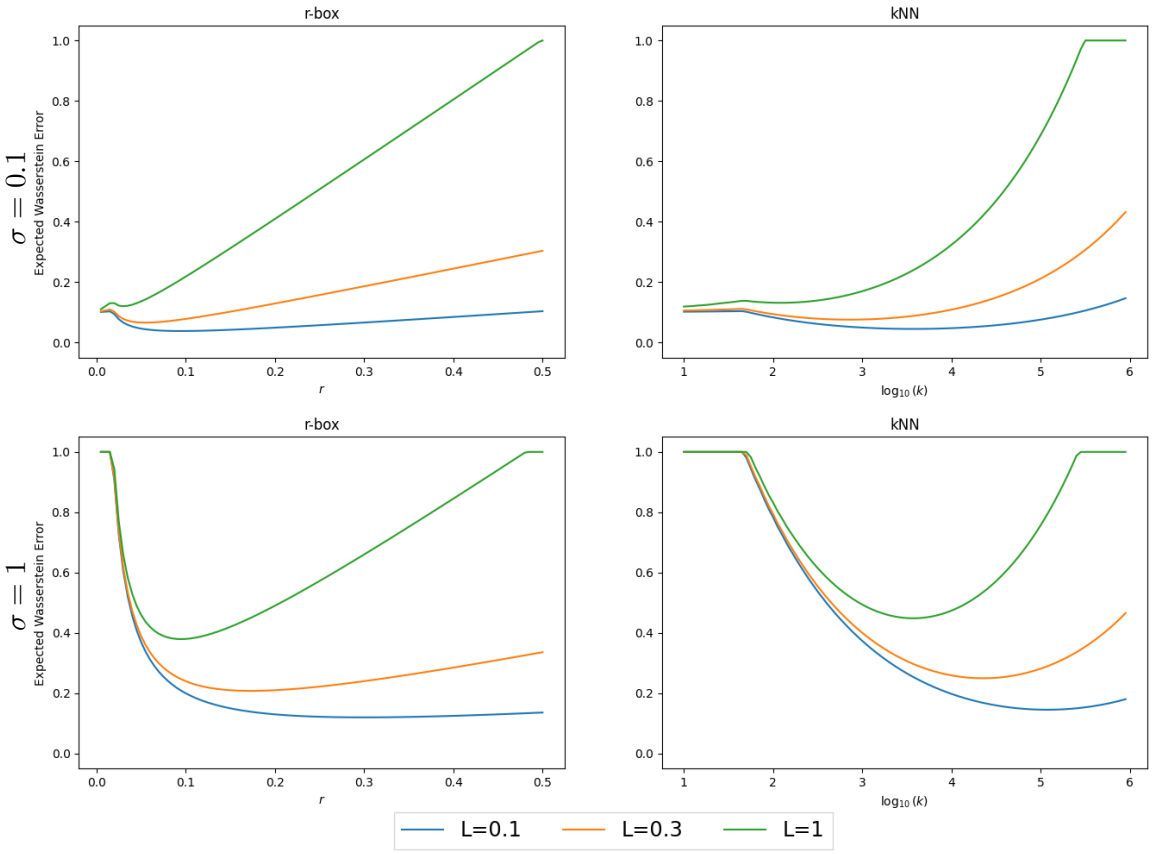


Figure 14: *Finite sample bounds with* $d_{\mathbb{X}} = d_{\mathbb{Y}} = 3$, $\overline{C} = \underline{c} = 1$, $M = 10^6$.
Note that the error is capped at 1, as the finite sample bounds are calculated for unit boxes. For the minimal values, we refer to Table 3.

Appendix B. Additional plots

In this section, we present additional plots to further support and complement the discussions provided earlier.

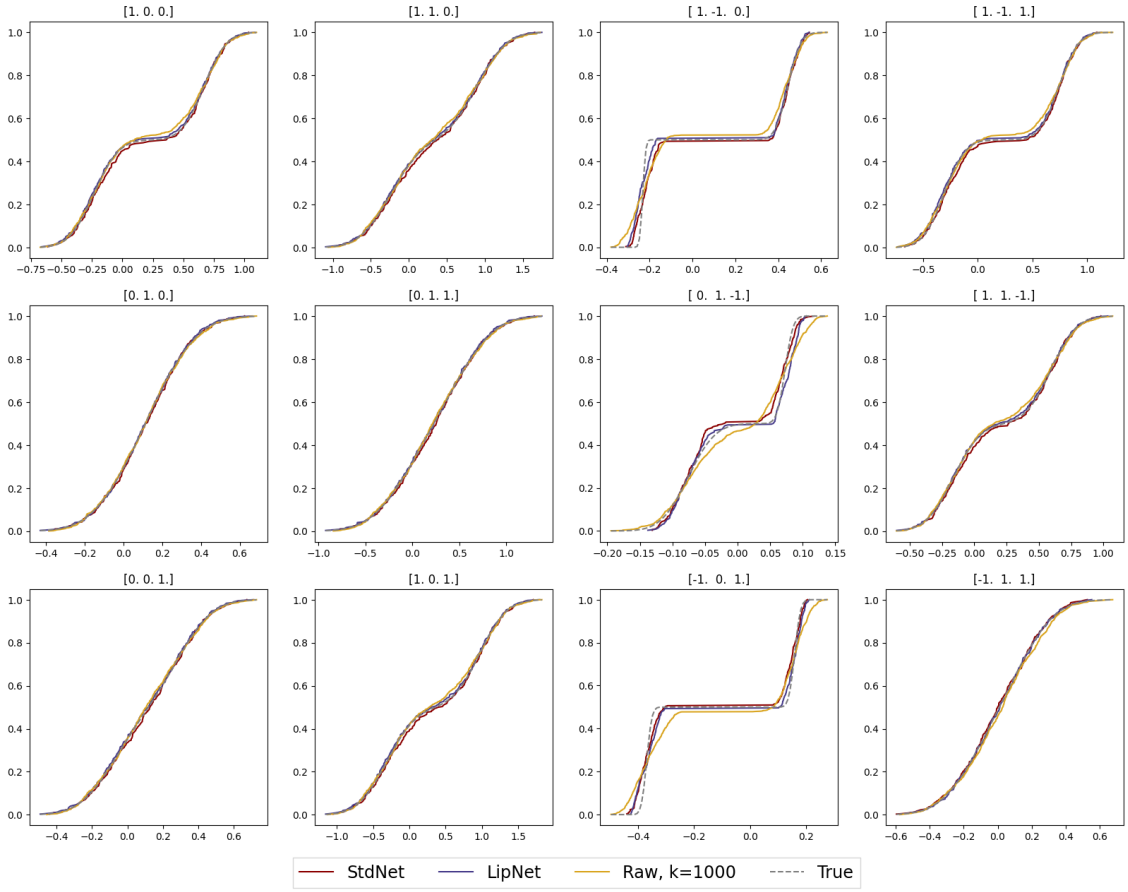


Figure 15: *Various estimators under Model 3, projections of conditional CDFs, $k = 1000$ for k-NN estimator.*

This follows the setting of Figure 5, expect that we set $k = 1000$ for the k-NN estimator plotted here. The neural estimator is trained under the same setting as that in Figure 5 Plot titles display the vectors used for projection. Note the difference in the x axis scale.

Appendix C. Additional Implementation Details

Table 4 below summarizes hyper-parameters of the neural networks. Additionally, we report the time consumption in Table 5 under various settings. All timings were obtained from a machine equipped with an Nvidia L40 GPU. It is important to note that the computational time does not fully reflect the curse of dimensionality, as the flexibility in choosing the hyper-parameter k allows for adjustments in practice. The appropriate selection of k, as discussed in Section A, is a nuanced issue and needs to be carefully re-evaluated based on the specific field information at hand.

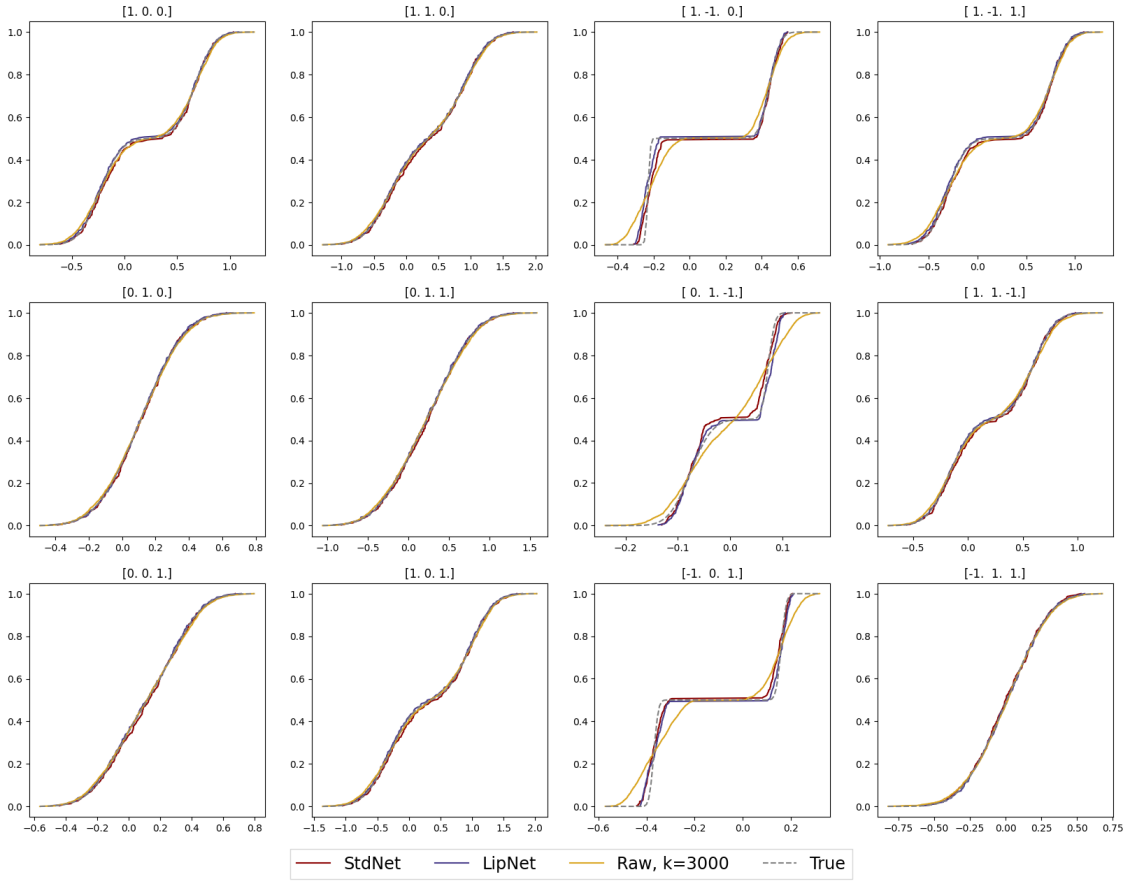


Figure 16: *Various estimators under Model 3, projections of conditional CDFs, $k = 3000$ for k-NN estimator.*

This follows the setting of Figure 5, expect that we set $k = 3000$ for the k-NN estimator plotted here. The neural estimator is trained under the same setting as that in Figure 5 Plot titles display the vectors used for projection. Note the difference in the x axis scale.

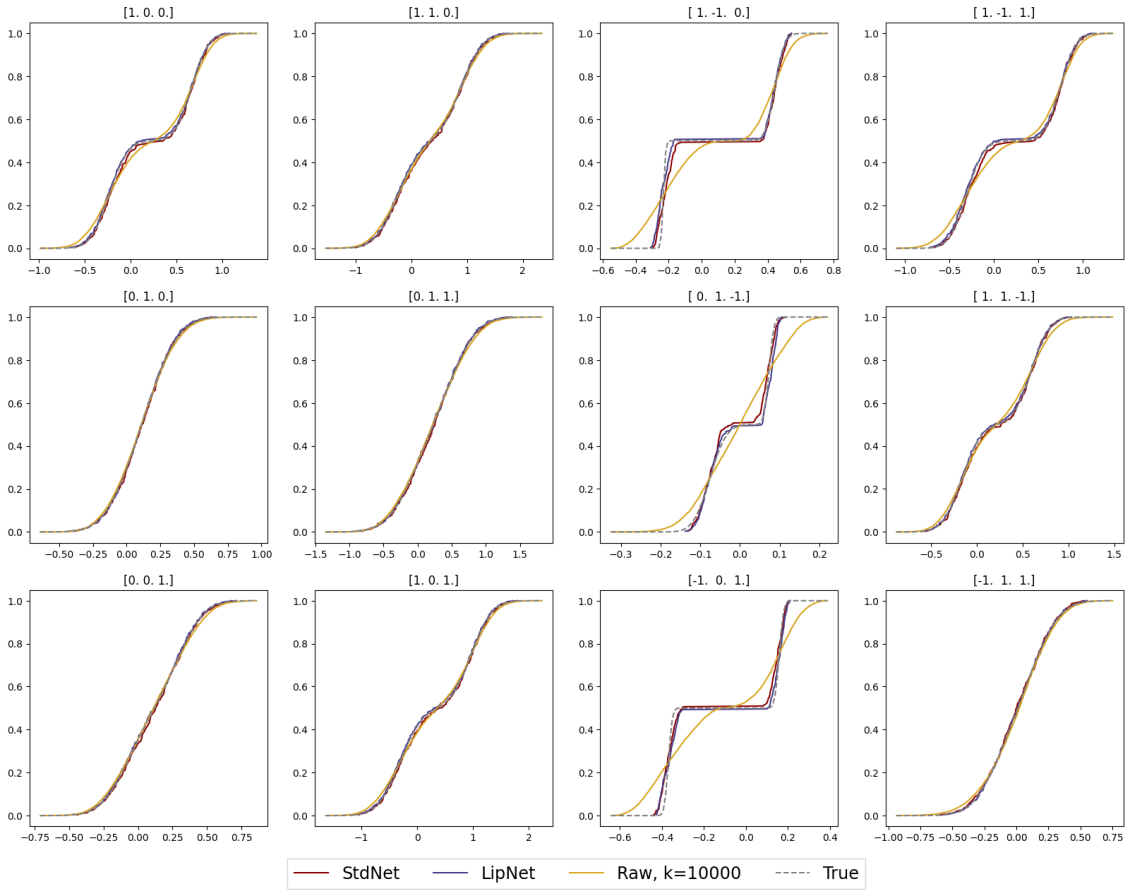


Figure 17: *Various estimators under Model 3, projections of conditional CDFs, $k = 10^4$ for k-NN estimator.*

This follows the setting of Figure 5, expect that we set $k = 10^4$ for the k-NN estimator plotted here. The neural estimator is trained under the same setting as that in Figure 5 Plot titles display the vectors used for projection. Note the difference in the x axis scale.

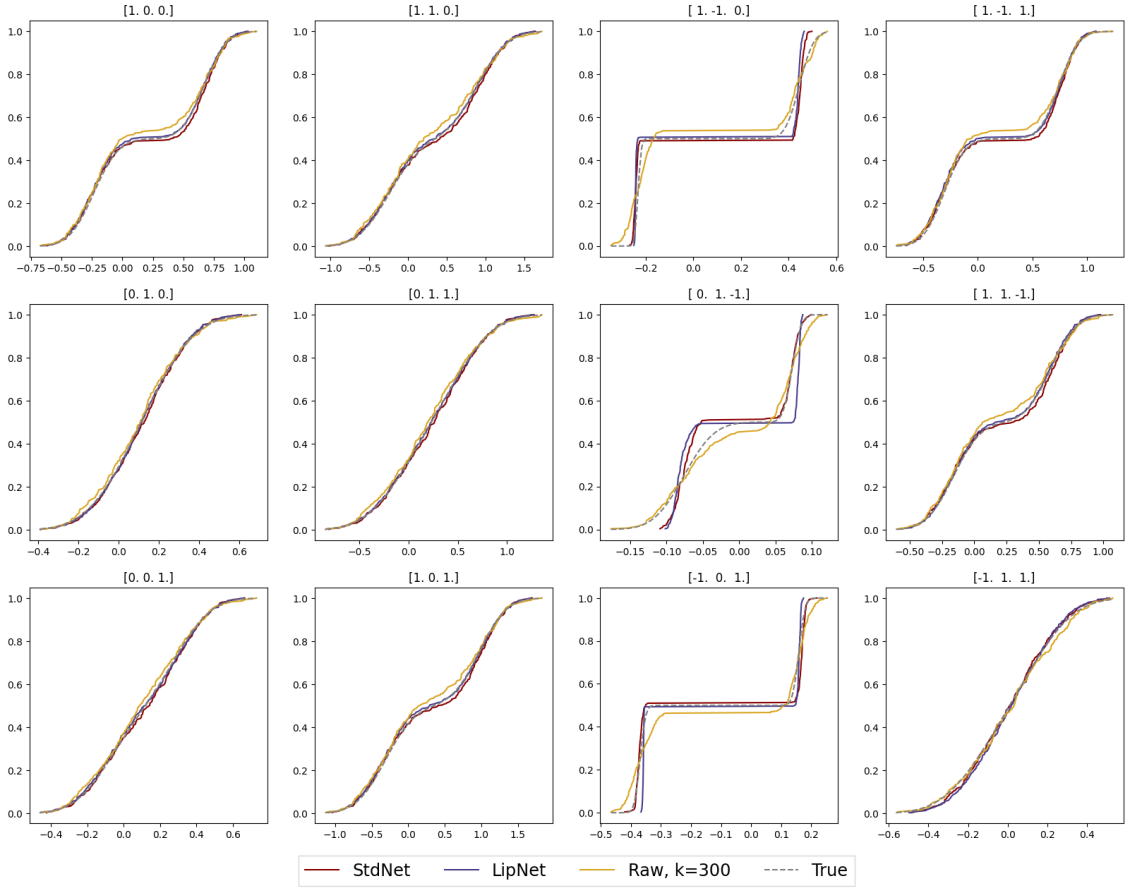
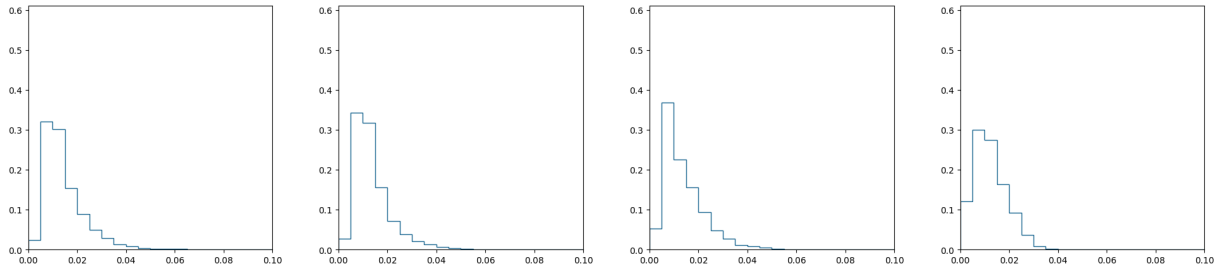


Figure 18: *Various estimators under Model 3, projections of conditional CDFs, both StdNet and Lipnet are trained without enforced sparsity.*

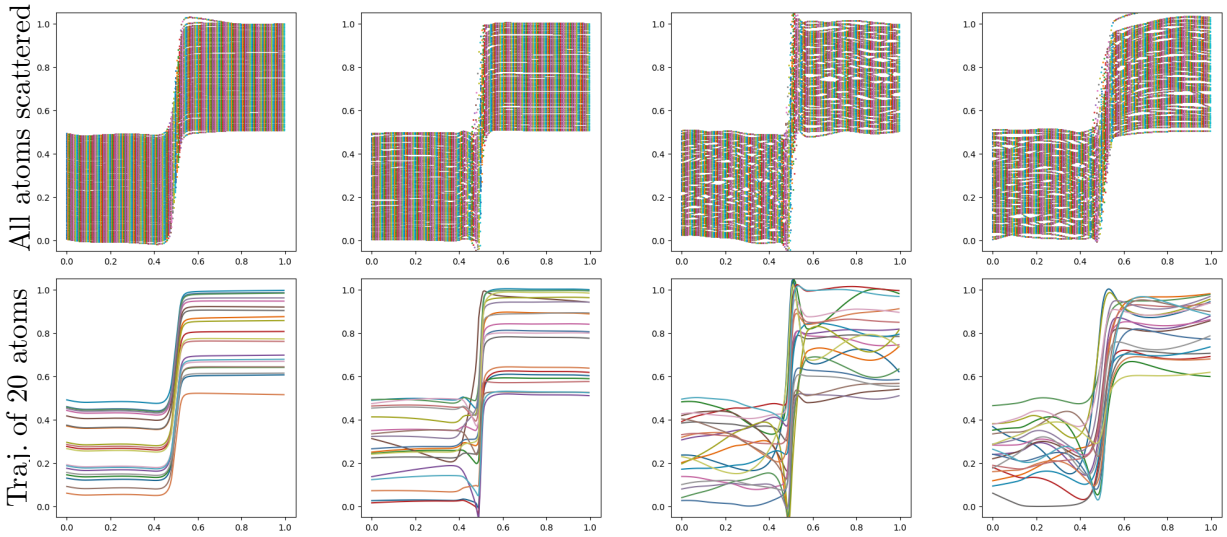
This follows the setting of Figure 5, expect that we do not enforce sparsity on the transport plan during the training of the neural estimator. Plot titles display the vectors used for projection. Note the difference in the x axis scale.



Raw $k = 1000$ *Raw* $k = 3,000$ *StdNet* *LipNet*

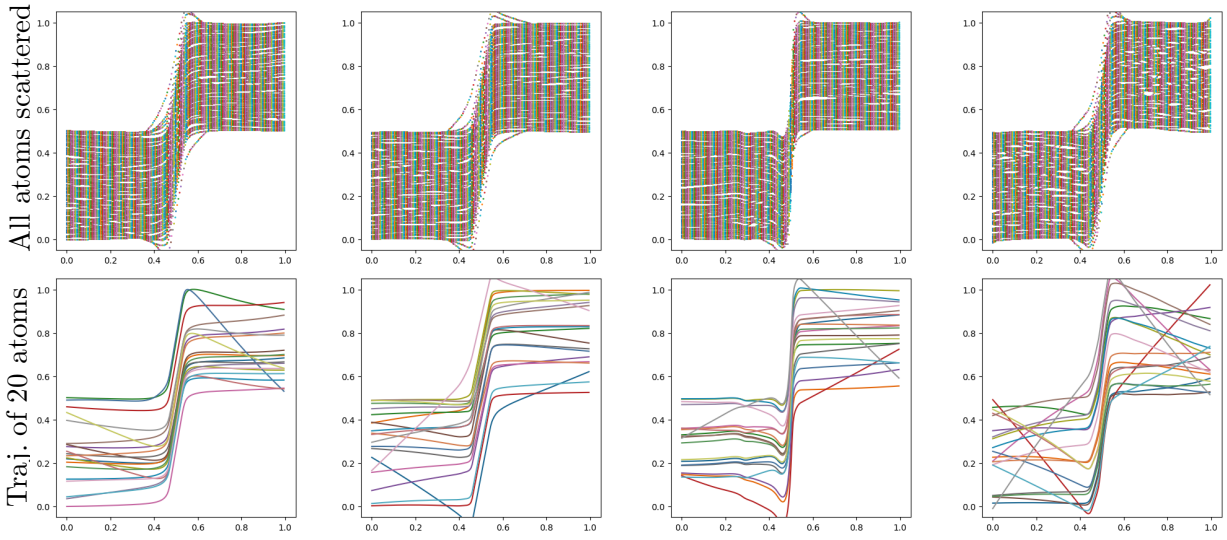
Figure 19: *Histogram of* $10,000$ *projected Wasserstein-*1 *errors, without enforced sparsity on transport plan.*

This follows the setting of Figure 6, expect that we do not enforce sparsity on the transport plan during the training of the neural estimator. Histograms for raw estimators remain the same. The histogram consists of 20 uniformly positioned bins between 0 to 0.1. The errors of different estimators are computed with the same set of query points and projection vectors. Errors larger than 0.1 will be placed in the right-most bins.



(a) $L = 0.01$ ***(b)*** $L = 0.03$ ***(c)*** $L = 1$ ***(d)*** $L = 3$

Figure 20: *LipNet under Model 2 with various L's.*



(a) $\tau = 10^{-1}$ ***(b)*** $\tau = 10^{-2}$ ***(c)*** $\tau = 10^{-5}$ ***(d)*** $\tau = 10^{-6}$

Figure 21: *LipNet under Model 2 with various τ's.*

Table 4 Hyper-parameters

Hyper-parameters	**Configuration**	**Note**
Sample size	1e4 for Model 1 & 2, 1e6 for Model 3	
k	100 for Model 1 & 2, 300 for Model 3	See Definition 9
Network stucture	StdNet: Layer-wise residual connection He et al. (2016), batch normalization (Ioffe and Szegedy (2015)) after affine transformation LipNet: Layer-wise residual connection (He et al. (2016)) with convex potential layer (Meunier et al. (2022))	
Input dimension	$d_{\mathbb{X}}$	
Output dimension	$d_{\mathbb{Y}} \times N_{\text{atom}}$	$N_{\text{atom}} = k$, see (10)
Number of hidden layers	5	
Number of neurons each hidden layer	$2k$	k as in Definition 9
Activation function	StdNet: ReLU LipNet: ELU	See Section 5.3.1
L	0.1	See (18)
τ	1e-3	See Algorithm 2
Optimizer	Adam (Kingma and Ba (2017)) with learning rate 10^{-3}	Learning rate is 10^{-2} for StdNet in Model 1 & 2
Batch size	100 for Model 1 & 2 256 for Model 3	
Number of episodes	5e3 for Model 1 & 2, 1e4 for Model 3	
RBSP setting	2^5 partition, 8 query points each part	See Section 3.1.1
Random bisecting ratio	$\sim$ Uniform($[0.45, 0.55]$)	See Section 3.1.1 and Algorithm 3
Ratio for mandatory slicing along the longest edge	5	See Section 3.1.1 and r_{edge} in Algorithm 3
Number of Sinkhorn iterations	5, if epoch ≤ 500 10, if epoch > 500	
ϵ	1, if epoch ≤ 100 0.1, if epoch $\in [100, 500]$ 0.05, if epoch > 500	See (33)
Enforced sparsity	Off, if epoch ≤ 500 On, if epoch > 500	See Section 5.2
γ	0.5	See (34)

Table 5 Computational times of 500 training steps for LipNet, with $M = 10^6$.

k	$d = 3$	$d = 10$
30	(24.6, 12.1)	(55.7, 16.3)
100	(27.0, 15.9)	(56.2, 16.9)
300	(29.2, 18.5)	(63.7, 24.7)
1000	(75.9, 64.8)	(151.3, 134.2)

This table presents the training time of LipNet for varying values of k and $d_{\mathbb{X}} = d_{\mathbb{Y}} = d$. For other hyper-parameters, please refer to Table 5. All times are reported in seconds. The first entry in each cell corresponds to the training time when using exact nearest neighbor search, while the second entry represents the time when employing ANNS-RBSP. The experiments were conducted on a machine equipped with an Nvidia L40 GPU.

Appendix D. Another set of results on fluctuation

D.1 On r-box estimator

Theorem 22 *Under Assumptions 2 and 3, and choosing r as in Theorem 7, let $\nu \in \mathcal{P}(\mathbb{X})$ be dominated by $\lambda_{\mathbb{X}}$ with constant $\overline{C} > 0$. Then, there is a constant $C > 0$ (which depends only on $d_{\mathbb{X}}, \underline{c}, \overline{C}$ and the constants involved in r), such that, for any $\varepsilon \geq 0$, we have*

$$\mathbb{P}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{P}_x^r\right) \mathrm{d}\nu(x) \geq \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{P}_x^r\right) \mathrm{d}\nu(x)\right] + \varepsilon\right] \leq \begin{cases} \exp\left(-CM^{\frac{2}{d_{\mathbb{X}}+2}}\varepsilon^2\right), & d_{\mathbb{Y}} = 1, 2, \\ \exp\left(-CM^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}\varepsilon^2\right), & d_{\mathbb{Y}} \geq 3. \end{cases}$$

Proof Let $\nu \in \mathcal{P}(\mathbb{X})$ as in the statement of the Theorem. We define

$$Z := \int_{\mathbb{X}} \mathcal{W}(P_x, \hat{P}_x^r) \, \mathrm{d}\nu(x),$$

and introduce the following discrete time filtration: $\mathscr{F}_0 := \{\emptyset, \Omega\}$ and $\mathscr{F}_m := \sigma(\bigcup_{i=1}^m \sigma(X_i, Y_i))$ for $m = 1, \ldots, M$. We consider the Doob's martingale $Z_m := \mathbb{E}\left[Z \,|\, \mathcal{F}_m\right]$, $m = 1, \ldots, M$. Note that $Z_M = Z$. We will apply Azuma-Hoeffding inequality (cf. (Wainwright, 2019, Corollary 2.20)) to complete the proof.

Let us define

$$\mathcal{D}^m := \{(X_1, Y_1), \ldots, (X_m, Y_m), (x_{m+1}, y_{m+1}), \ldots, (x_M, y_M)\}, \quad m = 1, \ldots, M, \tag{35}$$

$\mathcal{D}^0 := \{(x_\ell, y_\ell)\}_{\ell=0}^M$, and $\mathcal{D}^M := \mathcal{D}$. Note that, for all $m < M$, we have, by Assumptions 3 (i), conditional Fubini-Tonelli theorem, and independent lemma,

$$Z_m = \int_{\mathbb{X}} \int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \mathcal{W}\left(P_x, \hat{\mu}_{\mathcal{B}^r(x)}^{\mathcal{D}^m}\right) \bigotimes_{\ell=m+1}^{M} \psi(\mathrm{d}x_\ell \, \mathrm{d}y_\ell) \nu(\mathrm{d}x).$$

This together with the linearity of integral, the fact that ψ is a probability, and the triangular inequality of $\mathcal{W}$ implies that for $m = 1, \ldots, M$,

$$|Z_m - Z_{m-1}| \leq \int_{\mathbb{X}} \int_{(\mathbb{X}\times\mathbb{Y})^{M-m+1}} \mathcal{W}\left(\hat{\mu}_{\mathcal{B}^r(x)}^{\mathcal{D}^m}, \hat{\mu}_{\mathcal{B}^r(x)}^{\mathcal{D}^{m-1}}\right) \bigotimes_{\ell=m}^{M} \psi(\mathrm{d}x_\ell \, \mathrm{d}y_\ell) \nu(\mathrm{d}x). \tag{36}$$

Notice that, by definitions (1) and (35),

$$\left\{\hat{\mu}^{\mathsf{D}_m}_{\mathcal{B}^r(x)} \neq \hat{\mu}^{\mathsf{D}_{m-1}}_{\mathcal{B}^r(x)}\right\} \subseteq \left\{X_m \in \mathcal{B}^r(x)\right\} \cup \left\{x_m \in \mathcal{B}^r(x)\right\}. \tag{37}$$

Additionally, by definitions (1) and (35) again, on the event that $\left\{\hat{\mu}^{\mathsf{D}_m}_{\mathcal{B}^r(x)} \neq \hat{\mu}^{\mathsf{D}_{m-1}}_{\mathcal{B}^r(x)}\right\}$, we have

$$\mathcal{W}\left(\hat{\mu}^{\mathcal{D}^m}_{\mathcal{B}^r(x)}, \hat{\mu}^{\mathcal{D}^{m-1}}_{\mathcal{B}^r(x)}\right) \leq \left(1 + \sum_{\ell=1}^{m-1} \mathbb{1}_{\mathcal{B}^r(x)}(X_\ell) + \sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-1} \leq \left(1 + \sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-1} \tag{38}$$

Combining (36),(37), (38), and Fubini-Tonelli theorem, we get

$$\begin{aligned}
|Z_m - Z_{m-1}| &\leq \int_{\mathbb{X}} \int_{B(X_m,2r)\cup B(x_m,2r)} \int_{\mathbb{X}^{M+1-m}} \left(1 + \sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-1} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x)\xi(\mathrm{d}x_m) \\
&\leq \sup_{x_m\in\mathbb{X}} \int_{B(X_m,2r)\cup B(x_m,2r)} \int_{\mathbb{X}^{M+1-m}} \left(1 + \sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-1} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x) \qquad (39)
\end{aligned}$$

where the $2r$ in the domain of the integral stems from the usage of β^r in the definition of $\mathcal{B}^r$ (see Definition 5). Now, for fixed $x, x_m \in \mathbb{X}$, we have

$$\begin{aligned}
&\int_{\mathbb{X}^{M-m+1}} \left(1 + \sum_{\ell=m+1}^{M} \mathbb{1}_{\mathcal{B}^r(x)}(x_\ell)\right)^{-1} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell) \\
&= \sum_{\ell=0}^{M-m} \binom{M-m}{\ell} \xi\big(\mathcal{B}^r(x)\big)^\ell \Big(1 - \xi^r\big(\mathcal{B}^r(x)\big)\Big)^{M-m-\ell} (1+\ell)^{-1} \\
&= \frac{1}{(M-m+1)\xi\big(\mathcal{B}^r(x)\big)} \sum_{\ell=0}^{M-m} \binom{M-m+1}{\ell+1} \xi\big(\mathcal{B}^r(x)\big)^{\ell+1} \Big(1 - \xi\big(\mathcal{B}^r(x)\big)\Big)^{M-m-\ell} \\
&= \frac{1}{(M-m+1)\xi\big(\mathcal{B}^r(x)\big)} \sum_{\ell=1}^{M-m+1} \binom{M-m+1}{\ell} \xi\big(\mathcal{B}^r(x)\big)^{\ell} \Big(1 - \xi\big(\mathcal{B}^r(x)\big)\Big)^{M-m+1-\ell} \\
&= \frac{1 - \Big(1 - \xi\big(\mathcal{B}^r(x)\big)\Big)^{M-m+1}}{(M-m+1)\xi\big(\mathcal{B}^r(x)\big)} \leq 1 \wedge \big((M-m+1)\xi\big(\mathcal{B}^r(x)\big)\big)^{-1} \leq 1 \wedge \Big((M-m+1)\underline{c}(2r)^{d_{\mathbb{X}}}\Big)^{-1},
\end{aligned}$$

where we have used Assumption 3 (ii) in the last inequality. Recall $\overline{C}$ introduced in Theorem 22. In view of (39), we have

$$\begin{aligned}
|Z_m - Z_{m-1}| &\leq \sup_{x_m\in\mathbb{X}} \int_{B(X_m,2r)\cup B(x_m,2r)} 1 \wedge \Big((M-m+1)\underline{c}(2r)^{d_{\mathbb{X}}}\Big)^{-1} \nu(\mathrm{d}x) \\
&\leq 2\overline{C}(4r)^{d_{\mathbb{X}}} \left(1 \wedge \Big((M-m+1)\underline{c}(2r)^{d_{\mathbb{X}}}\Big)^{-1}\right) \\
&= \Big(\overline{C}2^{2d_{\mathbb{X}}+1} r^{d_{\mathbb{X}}}\Big) \wedge \frac{\overline{C}2^{d_{\mathbb{X}}+1}}{\underline{c}(M-m+1)} := C_m.
\end{aligned}$$

By Azuma-Hoeffding inequality (cf. (Wainwright, 2019, Corollary 2.20)), one obtains

$$\mathbb{P}(Z - \mathbb{E}[Z] \geq \varepsilon) \leq \exp\left(-\frac{2\varepsilon^2}{\sum_{m=1}^{M} C_m^2}\right).$$

To complete the proof, we substitute in the configuration of Theorem 7. Since we only aim to investigate the rate of $\sum_{m=1}^{M} C_m^2$ as $M \to \infty$, we simply set

$$r = M^{-\frac{1}{d_{\mathbb{X}}+d}} \quad \text{with} \quad d := 2 \vee d_{\mathbb{Y}}.$$

It follows that

$$\begin{aligned}\sum_{m=1}^{M} C_m^2 &\sim \sum_{m=1}^{M} M^{-\frac{2d_{\mathbb{X}}}{d_{\mathbb{X}}+d}} \wedge m^{-2} \\ &\lesssim \int_1^{\infty} M^{-\frac{2d_{\mathbb{X}}}{d_{\mathbb{X}}+d}} \wedge z^{-2}\,\mathrm{d}z \sim \int_1^{M^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+d}}} M^{-\frac{2d_{\mathbb{X}}}{d_{\mathbb{X}}+d}}\,\mathrm{d}z + \int_{M^{\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+d}}}^{\infty} z^{-2}\,\mathrm{d}z \sim M^{-\frac{d_{\mathbb{X}}}{d_{\mathbb{X}}+d}},\end{aligned}$$

which completes the proof. ■

D.2 On k-nearest-neighbor estimator

Theorem 23 *Under Assumptions 2 and 3, and the choice of k as in Theorem 10, there is a constant $C > 0$ (which depends only on $\underline{c}$ and the constants involved in k), such that, for any $\nu \in \mathcal{P}(\mathbb{X})$ and $\varepsilon \geq 0$, we have*

$$\mathbb{P}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right)\nu(\mathrm{d}x) \geq \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right)\nu(\mathrm{d}x)\right] + \varepsilon\right] \leq \begin{cases}\exp\left(-CM^{\frac{2}{d_{\mathbb{X}}+2}}\varepsilon^2\right), & d_{\mathbb{Y}} = 1, 2, \\ \exp\left(-CM^{\frac{d_{\mathbb{Y}}}{d_{\mathbb{X}}+d_{\mathbb{Y}}}}\varepsilon^2\right), & d_{\mathbb{Y}} \geq 3.\end{cases}$$

Proof [Proof of Theorem 23] For notational convenience, we will write $\hat{\mu}^{\mathsf{D}}_{\mathcal{N}^k(x)}$ for $\hat{\mu}^{\mathsf{D}}_{\mathcal{N}^{k,\mathsf{D}}(x)}$. Clearly, with $\mathsf{D} = \mathcal{D}$, we recover $\hat{\mu}^{\mathcal{D}}_{\mathcal{N}^k(x)} = \hat{\mu}^{\mathcal{D}}_{\mathcal{N}^{k,\mathcal{D}}(x)} = \check{P}_x^k$. In what follows, we let

$$Z := \int_{\mathbb{X}} \mathcal{W}\left(P_x, \check{P}_x^k\right)\nu(\mathrm{d}x).$$

We also define $\mathscr{F}_0 := \{\emptyset, \Omega\}$ and $\mathscr{F}_m := \sigma(\bigcup_{i=1}^m \sigma(X_i, Y_i))$ for $m = 1, \ldots, M$. The proof relies on an application of Azuma-Hoeffding inequality (cf. (Wainwright, 2019, Corollary 2.20)) to the Doob's martingale $\{\mathbb{E}[Z|\mathscr{F}_m]\}_{m=0}^M$. In order to proceed, we introduce a few more notations:

$$\begin{aligned}\boldsymbol{x} := (x_1, \ldots, x_M), \quad \mathcal{X} := (X_1, \ldots, X_M), \\ \mathcal{X}^m := (X_1, \ldots, X_m, x_{m+1}, \ldots, x_M), \\ \mathcal{D}^m := \{(X_1, Y_1), \ldots, (X_m, Y_m), (x_{m+1}, y_{m+1}), \ldots, (x_M, y_M)\}, \\ \eta_x^{k,\boldsymbol{x}} := \text{ the } k\text{-th smallest of } \{\|x_m - x\|_\infty\}_{m=1}^M.\end{aligned}$$

By independence lemma, we have

$$\begin{aligned}\mathbb{E}\left[Z\middle|\mathscr{F}_m\right] &= \int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{\mu}^{\mathcal{D}^m}_{\mathcal{N}^k(x)}\right)\nu(\mathrm{d}x) \bigotimes_{\ell=m+1}^{M} \psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell) \\ &= \int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \int_{\mathbb{X}} \mathcal{W}\left(P_x, \frac{1}{k}\left(\sum_{i=1}^{m} \mathbb{1}_{\|X_i - x\|_\infty \leq \eta_x^{k,\mathcal{X}^m}} \delta_{Y_i} + \sum_{\ell=m+1}^{M} \mathbb{1}_{\|x_\ell - x\|_\infty \leq \eta_x^{k,\mathcal{X}^m}} \delta_{y_\ell}\right)\right)\nu(\mathrm{d}x) \\ &\qquad \bigotimes_{\ell=m+1}^{M} \psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell),\end{aligned}$$

where we note that $(\mathbb{X}\times\mathbb{Y})^{M-m}$ and $\bigotimes_{\ell=m+1}^{M}\psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell)$ in the right hand side can be replaced by $(\mathbb{X}\times\mathbb{Y})^{M-m+1}$ and $\bigotimes_{\ell=m}^{M}\psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell)$ as the integrand is constant in x_m and ψ is a probability measure. Therefore, by Fubini's theorem and triangle inequality for $\mathcal{W}$, we have

$$\begin{aligned}&\left|\mathbb{E}\left[Z\middle|\mathscr{F}_m\right]-\mathbb{E}\left[Z\middle|\mathscr{F}_{m-1}\right]\right|\\&\quad\le\int_{\mathbb{X}}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m+1}}\mathcal{W}\left(\frac{1}{k}\left(\sum_{i=1}^{m}\mathbb{1}_{\|X_i-x\|_\infty\le\eta_x^{k,\mathcal{X}^m}}\delta_{Y_i}+\sum_{\ell=m+1}^{M}\mathbb{1}_{\|x_\ell-x\|_\infty\le\eta_x^{k,\mathcal{X}^m}}\delta_{y_\ell}\right)\right.\\&\qquad\left.\frac{1}{k}\left(\sum_{i=1}^{m-1}\mathbb{1}_{\|X_i-x\|_\infty\le\eta_x^{k,\mathcal{X}^{m-1}}}\delta_{Y_i}+\sum_{\ell=m}^{M}\mathbb{1}_{\|x_\ell-x\|_\infty\le\eta_x^{k,\mathcal{X}^{m-1}}}\delta_{y_\ell}\right)\right)\bigotimes_{\ell=m}^{M}\psi(\mathrm{d}x_\ell\,\mathrm{d}y_\ell)\,\mathrm{d}x.\end{aligned}\tag{40}$$

Above, the only difference between the two measures inside $\mathcal{W}$ is the m-th summand. Due to the definition of $\mathcal{W}$ and the boundedness of $\mathbb{X}$, the transport cost induced by altering the m-th summand is at most k^{-1}. It follows that

$$\left|\mathbb{E}\left[Z\middle|\mathscr{F}_m\right]-\mathbb{E}\left[Z\middle|\mathscr{F}_{m-1}\right]\right|\le\frac{1}{k},\quad m=1,\dots,M.\tag{41}$$

Below we further refine the upper bound of the absolute difference in the left hand side of (40) when $m=1,\dots,M-k$. For the integrand in the right hand side of (40) to be positive, it is necessary that

$$\mathbb{1}_{\|X_m-x\|_\infty\le\eta_x^{k,\mathcal{X}^m}}+\mathbb{1}_{\|x_m-x\|_\infty\le\eta_x^{k,\mathcal{X}^{m-1}}}\ge 1.$$

This, together with the tie breaking rule stipulated in Definition 9, further implies that

$$\mathbb{1}_{A_1^m}+\mathbb{1}_{A_2^m}\ge 1,$$

where

$$\begin{aligned}A_1^m&:=\left\{\text{at most }(k-1)\text{ of }x_\ell,\ell=m+1,\dots,M-m,\text{ falls into }B_x^{\|X_m-x\|_\infty}\right\},\\A_2^m&:=\left\{\text{at most }(k-1)\text{ of }x_\ell,\ell=m+1,\dots,M-m,\text{ falls into }B_x^{\|x_m-x\|_\infty}\right\}.\end{aligned}$$

Combining the above with the reasoning leading to (41), we yield

$$\begin{aligned}&\left|\mathbb{E}\left[Z\middle|\mathscr{F}_m\right]-\mathbb{E}\left[Z\middle|\mathscr{F}_{m-1}\right]\right|\\&\quad\le\frac{1}{k}\left(\int_{\mathbb{X}}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m}}\mathbb{1}_{A_1^m}\bigotimes_{\ell=m+1}^{M}\xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x)+\int_{\mathbb{X}}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m+1}}\mathbb{1}_{A_2^m}\bigotimes_{\ell=m}^{M}\xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x)\right)\end{aligned}$$

Above, we have replaced ψ in (40) by ξ because A_1^m and A_2^m no longer depend on $y_\ell,\ell=m+1,\dots,M$. The analogue applies to the domain of integral as well. We continue to have

$$\begin{aligned}&\left|\mathbb{E}\left[Z\middle|\mathscr{F}_m\right]-\mathbb{E}\left[Z\middle|\mathscr{F}_{m-1}\right]\right|\\&\quad\le\frac{1}{k}\left(\int_{\mathbb{X}}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m}}\mathbb{1}_{A_1^m}\bigotimes_{\ell=m+1}^{M}\xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x)+\int_{\mathbb{X}}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m+1}}\mathbb{1}_{A_2^m}\bigotimes_{\ell=m}^{M}\xi(\mathrm{d}x_\ell)\nu(\mathrm{d}x)\right)\\&\quad=:\frac{1}{k}(I_1^m+I_2^m).\end{aligned}\tag{42}$$

Regarding I_m^1 defined in (42), note that by Assumption 3,

$$\int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \mathbb{1}_{A_1^m} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell) = \mathbb{P}\left[\text{at most } (k-1) \text{ of } \check{X}_1, \ldots, \check{X}_{M-m} \text{ falls into } B_x^{\|x'-x\|_\infty}\right]\Big|_{x'=X_m},$$

where $\check{X}_1, \ldots, \check{X}_{M-m} \overset{\text{i.i.d.}}{\sim} \xi$. Below we define a CDF $G(r) := \underline{c} r^d, r \in [0, \underline{c}^{-\frac{1}{d}}]$. By Assumption 3 (ii), for any $x, x' \in \mathbb{X}$, we have

$$\int_{\mathbb{X}} \mathbb{1}_{\check{x}\in B_x^{\|x'-x\|_\infty}} \xi(\mathrm{d}\check{x}) \geq \underline{c} \int_{\mathbb{X}} \mathbb{1}_{\check{x}\in B_x^{\|x'-x\|_\infty}} \,\mathrm{d}\check{x} \geq \underline{c} \int_{\mathbb{X}} \mathbb{1}_{\check{x}\in B_{\mathbf{0}}^{\|x'-x\|_\infty}} \,\mathrm{d}\check{x} = G(\|x'-x\|_\infty),$$

where we have used the fact that $\|x'-x\|_\infty \leq 1 \leq \underline{c}^{-\frac{1}{d}}$ in the last equality. It follows from Lemma 17 that

$$\int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \mathbb{1}_{A_1^m} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell) \leq \sum_{j=0}^{k-1} \binom{M-m}{j} G(\|X_m - x\|_\infty)^j \big(1 - G(\|X_m - x\|_\infty)\big)^{M-m-j},$$

and thus, by letting $U \sim \mathrm{Uniform}(\mathbb{X})$,

$$\begin{aligned} I_1^m &\leq \int_{\mathbb{X}} \sum_{j=0}^{k-1} \binom{M-m}{j} G(\|X_m - x\|_\infty)^j \big(1 - G(\|X_m - x\|_\infty)\big)^{M-m-j} \,\mathrm{d}x \\ &= \mathbb{E}\left[\sum_{j=0}^{k-1} \binom{M-m}{j} G(\|x'-U\|_\infty)^j \big(1 - G(\|x'-U\|_\infty)\big)^{M-m-j}\right]\Big|_{x'=X_m}, \end{aligned}$$

where we note that the upper bounded no longer involves ν. For $x' \in \mathbb{X}$, it is obvious that

$$\mathbb{P}\big[\|x'-U\|_\infty \leq r\big] \geq \mathbb{P}\big[\|U\|_\infty \leq r\big], \quad r \in \mathbb{R},$$

i.e., $\|U\|_\infty$ stochastically dominates $\|x'-U\|_\infty$. Note additionally that, by Lemma 17 again, below is a non-decreasing function,

$$r \mapsto \sum_{j=0}^{k-1} \binom{M-m}{j} G(r)^j \big(1 - G(r)\big)^{M-m-j}.$$

Consequently,

$$I_1^m \leq \mathbb{E}\left[\sum_{j=0}^{k-1} \binom{M-m}{j} G(\|U\|_\infty)^j \big(1 - G(\|U\|_\infty)\big)^{M-m-j} \,\mathrm{d}x\right].$$

Since $\|U\|_\infty$ has CDF $r \mapsto r^{d_{\mathbb{X}}}, r \in [0,1]$ and $G(r) = \underline{c} r^d, r \in [0, \underline{c}^{-\frac{1}{d}}]$, we continue to obtain

$$\begin{aligned} I_1^m &\leq \sum_{j=0}^{k-1} \binom{M-m}{j} \int_{r=0}^{1} \underline{c} r^{d_{\mathbb{X}} j} (1 - \underline{c} r^{d_{\mathbb{X}}})^{M-m-j} \,\mathrm{d}r^{d_{\mathbb{X}}} \\ &\leq \underline{c}^{-1} \sum_{j=0}^{k-1} \frac{(M-m)!}{j!(M-m-j)!} \int_0^1 r^j (1-r)^{M-m-j} \,\mathrm{d}r. \end{aligned}$$

With a similar calculation as in (26), which involves beta distribution and gamma function, we arrive at

$$I_1^m \le \underline{c} \sum_{j=0}^{k-1} \frac{(M-m)!}{j!(M-m-j)!} \frac{j!(M-m-j)!}{(M-m+1)!} \le \frac{\underline{c}^{-1} k}{M-m}. \tag{43}$$

Regarding I_2^m defined in (42), we first let $\check{X}_0, \check{X}_1, \dots, \check{X}_{M-m} \overset{\text{i.i.d.}}{\sim} \xi$. Then, note that

$$\begin{aligned}\int_{(\mathbb{X}\times\mathbb{Y})^{M-m}} \mathbb{1}_{A_2^m} \bigotimes_{\ell=m+1}^{M} \xi(\mathrm{d}x_\ell) &\le \mathbb{P}\left[\text{at most } (k-1) \text{ of } \check{X}_1, \dots, \check{X}_{M-m} \text{ falls into } B_x^{\|\check{X}_0 - x\|_\infty}\right] \\ &\le \binom{M-m-1}{k-1} \binom{M-m}{k}^{-1} = \frac{k}{M-m},\end{aligned}$$

where the inequality in the second line is due to the symmetry stemming from Assumption 3 (i), and the fact that congestion along with the tie-breaking rule specified in Definition 9 may potentially rules out certain permutations. Consequently,

$$I_2^m \le \frac{k}{M-m}. \tag{44}$$

Putting together (41), (42), (43), and (44), we yield

$$\left|\mathbb{E}\left[Z\middle|\mathscr{F}_m\right] - \mathbb{E}\left[Z\middle|\mathscr{F}_{m-1}\right]\right| \le C_m := \frac{\overline{C}(\underline{c}^{-1}+1)}{M-m} \wedge \frac{1}{k}, \quad m = 1, \dots, M.$$

By Azuma-Hoeffding inequality (cf. (Wainwright, 2019, Corollary 2.20)),

$$\mathbb{P}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{\mu}^{\mathcal{D}}_{\mathcal{N}^k(x)}\right) \nu(\mathrm{d}x) - \mathbb{E}\left[\int_{\mathbb{X}} \mathcal{W}\left(P_x, \hat{\mu}^{\mathcal{D}}_{\mathcal{N}^k(x)}\right) \nu(\mathrm{d}x)\right] \ge \varepsilon\right] \le \exp\left(-\frac{\varepsilon^2}{2\sum_{m=1}^M C_m^2}\right), \ \varepsilon \ge 0.$$

To complete the proof, we substitute in the configuration of Theorem 10. Below we only investigate the rate of $\sum_{m=1}^M C_m^2$ as $M \to \infty$, and do not keep track of the constant. For simplicity, we set

$$k = k \sim M^{\frac{d}{d_{\mathbb{X}}+d}} \quad \text{with} \quad d := 2 \vee d_{\mathbb{Y}}$$

It follows that

$$\sum_{m=1}^{M} C_m^2 \sim \sum_{m=1}^{\lfloor M - M^{\frac{d}{d_{\mathbb{X}}+d}} \rfloor} \frac{1}{(M-m)^2} + \frac{M^{\frac{d}{d_{\mathbb{X}}+d}}}{M^{\frac{2d}{d_{\mathbb{X}}+d}}} \sim \int_{M^{\frac{d}{d_{\mathbb{X}}+d}}}^{\infty} \frac{1}{r^2}\,\mathrm{d}r + \frac{1}{M^{\frac{d}{d_{\mathbb{X}}+d}}} \sim M^{-\frac{d}{d_{\mathbb{X}}+d}},$$

which completes the proof. ∎

References

B. Acciaio and S. Hou. Convergence of adapted empirical measures on $\mathbb{R}^d$. arXiv:2211.10162, 2023.

B. Acciaio, A. Kratsios, and G. Pammer. Designing universal causal deep learning models: The geometric (hyper)transformer. Mathematical Finance, 34:671–735, 2024.

C. D. Aliprantis and K. C. Border. Infinite Dimensional Analysis: A Hitchhiker's Guide. Springer-Verlag Berlin Heidelberg, 2006.

F. Altekrüger, P. Hagemann, and G. Steidl. Conditional generative models are provably robust: Pointwise guarantees for bayesian inverse problems. Transactions on Machine Learning Research, 2023.

A. Araujo, A. J. Havens, B. Delattre, A. Allauzen, and B. Hu. A unified algebraic perspective on lipschitz neural networks. The Eleventh International Conference on Learning Representations, 2023.

J. Backhoff, D. Bartl, M. Beiglböck, and J. Wiesel. Estimating processes in adapted wasserstein distance. Annals of Applied Probability, 32:529–550, 2022.

T. Bai, J. Luo, Jun Zhao, B. Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, 2021.

P. L. Bartlett, D. J. Foster, and M. J. Telgarsky. Spectrally-normalized margin bounds for neural networks. Advances in Neural Information Processing Systems, 30, 2017.

D. M. Bashtannyk and R. J. Hyndman. Bandwidth selection for kernel conditional density estimation. Computational Statistics & Data Analysis, 36:279–298, 2001.

K. Bertin, C. Lacour, and V. Rivoirard. Adaptive pointwise estimation of conditional density function. Ann. Inst. H. Poincaré Probab. Statist., 52:939–980, 2016.

P. K. Bhattacharya and A. K. Gangopadhyay. Kernel and nearest-neighbor estimation of a conditional quantile. The Annals of Statistics, 18:1400–1415, 1990.

A. Bhowmick, M. D'Souza, and G. S. Raghavan. Lipbab: Computing exact lipschitz constant of relu networks. International Conference on Artificial Neural Networks, 2021.

G. Biau and L. Devroye. Lectures on the Nearest Neighbor Method. Springer, 2015.

V. I. Bogachev. Measure Theory Volume II. Springer-Verlag Berlin Heidelberg, 2007.

J. Booth, P. Hall, and A. Wood. Bootstrap estimation of conditional distributions. The Annals of Statistics, 20:1594–1610, 1992.

P. Bountakas, A. Zarras, A. Lekidis, and C. Xenakis. Defense strategies for adversarial machine learning: A survey. Computer Science Review, 49, 2023.

Z. Cheng and S. Jaimungal. Distributional dynamic risk measures in markov decision processes. arXiv:2203.09612, 2023.

Y. Chow, A. Tamar, S. Mannor, and M. Pavone. Risk-sensitive and robust decision-making: a cvar optimization approach. Advances in Neural Processing Systems, 2015.

D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). arXiv:1511.07289, 2016.

Anthony Coache, Sebastian Jaimungal, and Álvaro Cartea. Conditionally elicitable dynamic risk measures for deep reinforcement learning. SIAM Journal on Financial Mathematics, 14(4):1249–1289, 2023.

J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. Proceedings of the 36th International Conference on Machine Learning, 97, 2019.

E. Demirkayaa, Y. Fan, L. Gao, J. Lv, P. Vossler, and J. Wang. Optimal nonparametric inference with two-scale distributional nearest neighbors. Journal of the American Statistical Association, 199:297–307, 2024.

L. Devroye. Necessary and sufficient conditions for the pointwise convergence of nearest neighbor regression function estimates. Probability Theory and Related Fields, 61:467–481, 1982.

R. M. Dudley. The speed of mean glivenko-cantelli convergence. The Annals of Mathematical Statistics, 40(1):40–50, 1969.

J. Fan. Design-adaptive nonparametric regression. Journal of the American Statistical Association, 87:998–1004, 1992.

M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. Advances in Neural Information Processing Systems, 32, 2019.

M. Fazlyab, T. Entesari, A. Roy, and R. Chellappa. Certified robustness via dynamic margin maximization and improved lipschitz regularization. Advances in Neural Information Processing Systems, 36, 2024.

F. Ferraty and P. Vieu. Nonparametric Functional Data Analysis: Theory and Practics. Springer, 2006.

E. Fetaya, J., W. Grathwohl, and R. Zemel. Understanding the limitations of conditional generative models. International Conference on Learning Representations, 2020.

E. Fix and J. L. Hodges. Discriminatory analysis; nonparametric discrimination, consistency properties. USAF SAM Series in Statistics, Project No. 21-49-004, 1951.

N. Fournier. Convergence of the empirical measure in expected wasserstein distance: non-asymptotic explicit bounds in $\mathbb{R}^d$. ESAIM: Probability and Statistics, 27:749–775, 2023.

Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. Probability Theory and Related Fields, 162:707–738, 2015.

T. Gasser and H. Müller. Kernel estimation of regression functions. Smoothing Technique for Curve Estimation, pages 23–63, 1979.

M. Giegrich, R. Oomen, and C. Reisinger. K-nearest-neighbor resampling for off-policy evaluation in stochastic control. arXiv:2306.04836, 2024.

I. Gijbels and A. Goderniaux. Bandwidth selection for changepoint estimation in nonparametric regression. Technometrics, 46:76–86, 2004.

H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree. Regularisation of neural networks by enforcing lipschitz continuity. Machine Learning, 110:393–416, 2021.

L. Györfi, M. Kohler, A. Krzyźak, and H. Walk. A Distribution-Free Theory of Nonparametric Regression. Springer, 2002.

K. Hajebi, Y. Abbasi-Yadkori, H. Shahbazi, and H. Zhang. Fast approximate nearest-neighbor search with k-nearest neighbor graph. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 2011.

P. Hall and J. D. Hart. Nonparametric regression with long-range dependence. Stochastic Processes and their Applications, 36:339–351, 1990.

P. Hall, R. C. L. Wolff, and Q. Yao. Methods for estimating a conditional distribution function. Journal of the American Statistical Association, 94:154–163, 1999.

W. Hardle and J. S. Marron. Optimal bandwidth selection in nonparametric regression function estimation. The Annals of Statistics, 13:1465–1481, 1985.

K. He, X. Zhang, S., and J. Sun. Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016.

Ruiyang Hong and Anastasis Kratsios. Bridging the gap between approximation and learning via optimal approximation by relu mlps of maximal regularity. arXiv:2409.12335, 2024.

B. Hosseini, A. W. Hsu, and A. Taghvaei. Conditional optimal transport on function spaces. arXiv:2311.05672, 2024.

S. Hou. Convergence of the adapted smoothed empirical measures. arXiv:2401.14883, 2024.

W. Huang and W. B. Haskell. Risk-aware q-learning for markov decision processes. IEEE 56th Annual Conference on Decision and Control, 2017.

Y. Huang, H. Zhang, Y. Shi, J. Z. Kolter, and A. Anandkumar. Training certifiably robust neural networks with efficient local lipschitz bounds. Advances in Neural Information Processing Systems, 34, 2021.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. Proceedings of the 32nd International Conference on Machine Learning, pages 448–456, 2015.

M. Jordan and A. G. Dimakis. Exactly computing the local lipschitz constant of relu networks. Advances in Neural Information Processing Systems, 33, 2021.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv:1412.6980, 2017.

Benoît R. Kloeckner. Empirical measures: regularity is a counter-curse to dimensionality. ESAIM: Probability and Statistics, 24:408–434, 2020.

M. Kohler, A. Krzyżak, and H. Walk. Optimal global rates of convergence for nonparametric regression with unbounded data. Journal of Statistical Planning and Inference, 193:1286–1296, 2009.

M. Kohler, A. Krzyżak, and H. Walk. Uniform convergence rate for nonparametric regrssion and principle component analysis with functional/longtitudeinal data. The Annals of Statistics, 38: 3321–3351, 2010.

M. Köhler, A. Schindler, and S. Sperlich. A review and comparison of bandwidth selection methods for kernel regression. International Statistical Review, 82:243–274, 2014.

A. Kratsios. Universal regular conditional distributions via probabilistic transformers. Constructive Approximation, 57:1145–1212, 2023.

C. Lacour. Adaptive pointwise estimation of conditional density function. Ann. Inst. H. Poincaré Probab. Statist., 43:571–597, 2007.

Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Mingjie Li, Wenjie Zhang, and Xuemin Lin. Approximate nearest neighbor search on high dimensional data — experiments, analyses, and improvement. IEEE Transactions on Knowledge and Data Engineering, 32:1475–1488, 2020a.

Y. Li, S. Akbar, and J. Oliva. Acflow: Flow models for arbitrary conditional likelihoods. Proceedings of the 37th International Conference on Machine Learning, 119:5831–5841, 2020b.

H. D. Liu, F. Williams, A. Jacobson, S. Fidler, and O. Litany. Learning smooth neural functions via lipschitz regularization. SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference, 2022.

Y. P. Mack. Local properties of k-nn regression estimates. SIAM Journal on Algebraic Discrete Methods, 2:311–323, 1981.

F. Martínez, M. P. Frías, M. D. Pérez, and A. J. Rivera. A methodology for applying k-nearest neighbor to time series forecasting. Artificial Intelligence Review, 52:2019–2037, 2017.

Milan Merkle. Inequalities for the gamma function via convexity. Advances in Inequalities for Special Functions, pages 81–100, 2008.

L. Meunier, B. J. Delattre, A. Araujo, and A. Allauzen. A dynamical system perspective for lipschitz neural networks. International Conference on Machine Learning, 165, 2022.

M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv:1411.1784, 2014.

K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf. Kernel mean embedding of distributions: A review and beyond. Foundations and Trends in Machine Learning, 10:1–144, 2017.

E. A. Nadaraya. On estimating regression. Theory of Probability and Its Applications, 1964.

Bao Nguyen, Binh Nguyen, Hieu Trung Nguyen, and Viet Anh Nguyen. Generative conditional distributions by neural (entropic) optimal transport. Proceedings of the 41st International Conference on Machine Learning, 235:37761–37775, 2024.

O. H. M. Padilla, J. Sharpnack, Y. Chen, and D. M. Witten. Adaptive nonparametric regression with the k-nearest neighbour fused lasso. Technometrics, 107:293–310, 2020.

G. Papamakarios, T. Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. Advances in Neural Information Processing Systems, 30, 2017.

P. Pauli, A. Koch, J. Berberich, P. Köhler, and F. Allgöwer. Training robust neural networks using lipschitz bounds. IEEE Control Systems Letters, 6:121–126, 2022.

Gabriel Peyré and Marco Cuturi. Computational Optimal Transport: With Applications to Data Science. Foundations and Trends in Machine Learning, 2019.

G. Ch. Pflug and A. Pichler. From empirical observations to tree models for stochastic optimization: Convergence properties. SIAM Journal on Optimization, 26:1715–1740, 2016.

M. Rachdi, A. Laksaci, Z. Kaid, A. Benchiha, and F. A. Al-Awadhi. k-nearest neighbors local linear regression for functional and missing data at random. Statistica Neerlandica, 75, 2021.

P. Ram and K. Sinha. Revisiting kd-tree for nearest neighbor search. KDD'19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, page 1378–1388, 2019.

D. Rudolf and N. Schweizer. Perturbation theory for markov chains via wasserstein distance. Bernoulli, 24:2610–2639, 2018.

J. J. Ryu and Y. Kim. Minimax regression via adaptive nearest neighbor. arXiv:2202.02464, pages 1447–1451, 2022.

D. Scott. Multivariate Density Estimation: Theory, Practics, and Visualization. Wiley, 2015.

D. Shah and Q. Xie. Q-learning with nearest neighbors. Advances in Neural Information Processing Systems, 31, 2010.

Z. Shi, Y. Wang, H. Zhang, J. Z. Kolter, and Cho-Jui Hsieh. Efficiently computing local lipschitz constants of neural networks via bound propagation. Advances in Neural Information Processing Systems, 35, 2022.

J. S. Simonoff. Smoothing Methods in Statistics. Springer, 1996.

S. Singla, S. Singla, and S. Feizi. Improved deterministic l2 robustness on cifar-10 and cifar-100. The Tenth International Conference on Learning Representations, 2022.

C. J. Stone. Optimal global rates of convergence for nonparametric regression. The Annals of Statistics, 10:1040–1053, 1982.

A. Trockman and J. Z. Kolter. Orthogonalizing convolutional layers with the cayley transform. International Conference on Learning Representations, 36, 2021.

Y. Tsuzuku, I. Sato, and M. Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. Advances in Neural Information Processing Systems, 31, 2018.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. Advances in Neural Information Processing Systems, 30, 2017.

Cédric Villani. Optimal transport: Old and new, volume 338. Springer Science & Business Media, 2008.

A. Virmaux and K. Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. Advances in Neural Information Processing Systems, 31, 2018.

M. Šmíd and V. Kozmík. Approximation of multistage stochastic programming problems by smoothed quantization. Review of Managerial Science, 2024.

M. Vuletić, F. Prenzel, and M. Cucuringu. Fin-gan: forecasting and classifying financial time series via generative adversarial networks. Quantitative Finance, 24, 2024.

M. J. Wainwright. High-dimensional statistics: A non-asymptotic viewpoint, volume 48. Cambridge university press, 2019.

R. Wang and I. Manchester. Direct parameterization of lipschitz-bounded deep networks. Proceedings of the 40th International Conference on Machine Learning, 202, 2023.

L. Wasserman. All of Nonparametric Statistics. Springer, 2006.

G. S. Watson. Smooth regression analysis. Sankhya: The Indian Journal of Statistics, Series A, 26: 359–372, 1964.

T. Xu and B. Acciaio. Conditional cot-gan for video prediction with kernel smoothing. NeurIPS 2022 Workshop on Robustness in Sequence Modeling, 2022.

A. Xue, L. Lindemann, A. Robey, H. Hassani, G. J. Pappas, and R. Alur. Chordal sparsity for lipschitz constant estimation of deep neural networks. 2022 IEEE 61st Conference on Decision and Control, 6:3389–3396, 2022.

B. Zhang, D. Jiang, D. He, and L. Wang. Rethinking lipschitz neural networks and certified robustness: A boolean function perspective. Advances in Neural Information Processing Systems, 35, 2022.

P. Zhao and L. Lai. Minimax regression via adaptive nearest neighbor. 2019 IEEE International Symposium on Information Theory, pages 1447–1451, 2019.

W. Zhao and E. G. Tabak. Adaptive kernel conditional density estimation. 2023.