Electronic Journal of Statistics
Vol. 11 (2017) 2800–2831
ISSN: 1935-7524
DOI: 10.1214/17-EJS1302

Converting high-dimensional regression to high-dimensional conditional density estimation*

Rafael Izbicki†

Department of Statistics, Federal University of São Carlos, Brazil
e-mail: rafaelizbicki@gmail.com

and

Ann B. Lee‡

Department of Statistics, Carnegie Mellon University, USA
e-mail: annlee@cmu.edu

Abstract: There is a growing demand for nonparametric conditional density estimators (CDEs) in fields such as astronomy and economics. In astronomy, for example, one can dramatically improve estimates of the parameters that dictate the evolution of the Universe by working with full conditional densities instead of regression (i.e., conditional mean) estimates. More generally, standard regression falls short in any prediction problem where the distribution of the response is more complex with multimodality, asymmetry or heteroscedastic noise. Nevertheless, much of the work on high-dimensional inference concerns regression and classification only, whereas research on density estimation has lagged behind. Here we propose *FlexCode*, a fully nonparametric approach to conditional density estimation that reformulates CDE as a non-parametric orthogonal series problem where the expansion coefficients are estimated by regression. By taking such an approach, one can efficiently estimate conditional densities and not just expectations in high dimensions by drawing upon the success in high-dimensional regression. Depending on the choice of regression procedure, our method can adapt to a variety of challenging high-dimensional settings with different structures in the data (e.g., a large number of irrelevant components and nonlinear manifold structure) as well as different data types (e.g., functional data, mixed data types and sample sets). We study the theoretical and empirical performance of our proposed method, and we compare our approach with traditional conditional density estimators on simulated as well as real-world data, such as photometric galaxy data, Twitter data, and line-of-sight velocities in a galaxy cluster.

MSC 2010 subject classifications: Primary 62G07, 62G15; secondary 62G08.

*We thank Michelle Ntampaka and Hy Trac for sharing the data for the galaxy cluster mass example, and Peter E. Freeman for his help with the photometric redshift and galaxy cluster mass studies.

†Partially supported by *Fundação de Amparo à Pesquisa do Estado de São Paulo* (2014/25302-2 and 2017/03363-8)

‡Partially supported by NSF DMS-1520786, and the National Institute of Mental Health grant R37MH057881

Keywords and phrases: Nonparametric inference, conditional density, high-dimensional data, prediction intervals, functional conditional density estimation.

Received July 2016.

Contents

1. Introduction

A challenging problem in modern statistical inference is how to estimate a conditional density of a random variable $Z \in \mathbb{R}$ given a high-dimensional random vector $\mathbf{X} \in \mathbb{R}^D$, $f(z|\mathbf{x})$. This quantity plays a key role in several statistical problems in the sciences where the regression function $\mathbb{E}[Z|\mathbf{x}]$ is not informative enough due to multi-modality and asymmetry of the conditional density.

For example, several recent works in cosmology [49, 32, 44, 18] have shown that one can significantly reduce systematic errors in cosmological analyses by using the full probability distribution of photometric redshifts z (a key quantity that relates the distance of a galaxy to the observer) given galaxy colors $\mathbf{x}$ (i.e., differences of brightness measures at two different wavelengths). Other fields where conditional density estimation plays a key role are time series forecasting in economics [31] and approximate Bayesian methods [14, 29, 42]. Conditional densities can also be used to construct accurate predictive intervals for new

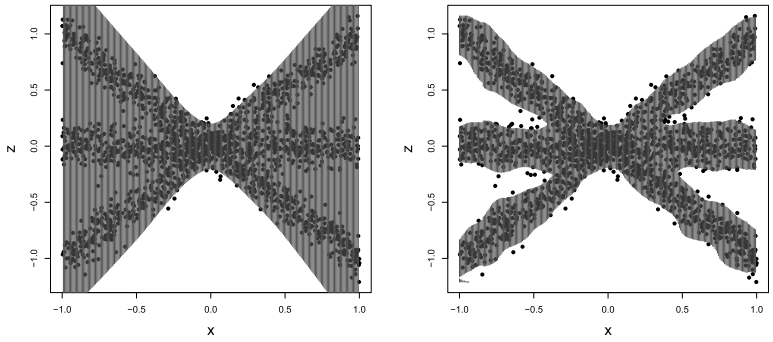


FIG 1. *A toy example where nonparametric regression methods fail to capture the underlying structure and return too wide prediction bands, whereas a nonparametric conditional density estimator automatically returns informative predictive bands. The left plot shows 95% predictions bands from local linear regression, and the right plot shows 95% highest predictive density (HPD) bands derived from* FlexCode-SAM *estimates of the conditional density.*

observations in settings with complicated sources of errors [15] or multimodal distributions (see Fig. 1 and Fig. 7 for examples).

Nevertheless, whereas a large literature has been devoted to estimating the regression $\mathbb{E}[Z|\mathbf{x}]$, statisticians have paid far less attention to estimating the full conditional density $f(z|\mathbf{x})$, especially when $\mathbf{x}$ is high-dimensional. Most attempts to estimate $f(z|\mathbf{x})$ can effectively only handle up to about 3 covariates (see, e.g., Fan et al. [13]). In higher dimensions, such methods typically rely on a prior dimension reduction step which, as is the case with any data reduction, can result in significant loss of information.

Contribution. There is currently no *general* procedure for converting successful regression estimators (that is, estimators of the conditional mean $\mathbb{E}[Z|\mathbf{x}]$) to estimators of the full conditional density $f(z|\mathbf{x})$ — indeed, this is a nontrivial problem. In this paper, we propose a fully nonparametric approach to conditional density estimation, which reformulates CDE as an orthogonal series problem where the expansion coefficients are estimated by regression. By taking such an approach, one can efficiently estimate conditional densities in high dimensions by drawing upon the success in high-dimensional regression. Depending on the choice of regression procedure, our method can exploit different types of sparse structure in the data, as well as handle different types of data.

For example, in a setting with submanifold structure, our estimator adapts to the intrinsic dimensionality of the data with a suitably chosen regression method; such as, nearest neighbors, local linear, tree-based or spectral series regression [4, 33, 34, 36]. Similarly, if the number of relevant covariates (i.e., covariates that affect the distribution of Z) is small, one can construct a good conditional density estimator using lasso, SAM, Rodeo or other additive-based regression estimators [54, 35, 39, 57]. Because of the flexibility of our approach, the method is able to overcome the the curse of dimensionality in a variety

of scenarios with faster convergence rates and better performance than traditional conditional density estimators; see Sections 3–4 for specific examples and analysis. By choosing appropriate regression methods, the method can also handle different types of covariates that represent discrete data, mixed data types, functional data, circular data, and so on, which generally require hand-tailored techniques (e.g., Di Marzio et al. [8]). Most notably, Sec. 3.4 describes an entirely new area of conditional density estimation (here referred to as "Distribution CDE") where a predictor is an entire *sample set* from an underlying distribution.

We call our general approach *FlexCode*, which stands for *Flex*ible nonparametric *co*nditional *d*ensity *e*stimation via regression.

Existing Methodology. With regards to existing methods for estimating $f(z|\mathbf{x})$, several nonparametric estimators have been proposed when $\mathbf{x}$ lies in a *low-dimensional* space. Many of these methods are based on first estimating $f(z, \mathbf{x})$ and $f(\mathbf{x})$ with, for example, kernel density estimators [48], and then combining the estimates according to $f(z|\mathbf{x}) = \frac{f(z,\mathbf{x})}{f(\mathbf{x})}$. Several works further improve upon such an approach by using different criteria and shortcuts to tune parameters as well as creating fast shortcuts to implement these methods (e.g., Hyndman et al. [26], Ichimura and Fukuda [27]). Other approaches to conditional density estimation in low dimensions include using locally polynomial regression [12], least squares approaches [51] and density estimation through quantile estimation [53]; see Bertin et al. [3] and references therein for other methods.

For moderate dimensions, Hall et al. [20] propose a method for tuning parameters in kernel density estimators which automatically determines which components of $\mathbf{x}$ are relevant to $f(z|\mathbf{x})$. The method produces good results but is not practical for high-dimensional data sets: Because it relies on choosing a different bandwidth for each covariate, it has a high computational cost that increases with both the sample size n and the dimension D, with prohibitive costs even for moderate n's and D's. Similarly, Shiga et al. [50] propose a conditional estimator that selects relevant components but under the restrictive assumption that $f(z|\mathbf{x})$ has an additive structure; moreover the method scales as $O(D^3)$, which is also computationally prohibitive for moderate dimensions. Another framework is developed by Efromovich [10], who proposes an orthogonal series estimator that automatically performs dimension reduction on $\mathbf{x}$ when several components of this vector are conditionally independent of the response. Unfortunately, the method requires one to compute $D+1$ tensor products, which quickly becomes computationally intractable even for as few as 10 covariates. More recently, Izbicki and Lee [28] propose an alternative orthogonal series estimator that uses a basis that adapts to the geometry of the data. They show that their approach, called Spectral Series CDE, as well as the k-nearest neighbor method by Zhao and Liu [59], work well in high dimensions when there is submanifold structure. These methods, however, do not perform well when $\mathbf{x}$ has irrelevant components.

FlexCode, on the other hand, is flexible enough to overcome the difficulties of other methods under a large variety of situations because it makes use of

the many existing regression methods for high-dimensional inference. As an example, Fig. 2 shows the level sets of the estimated conditional density in a challenging problem that involves ≈500 covariates. Here we estimate $f(\mathbf{z}|\mathbf{x})$, where $\mathbf{x}$ is the content of a tweet and $\mathbf{z}$ is the location where it was posted (latitude and longitude). FlexCode, based on sparse additive regression, is able to estimate the location of tweets even in ambiguous cases (there is a Long Beach both in California and in Connecticut, which is reflected by the results in the bottom right plot in Fig. 2); we are not aware of any other existing fully nonparametric method that are able to estimate this quantity with reasonable precision as well as attach meaningful measures of uncertainty. See more details about this example in Sec. 3.3.

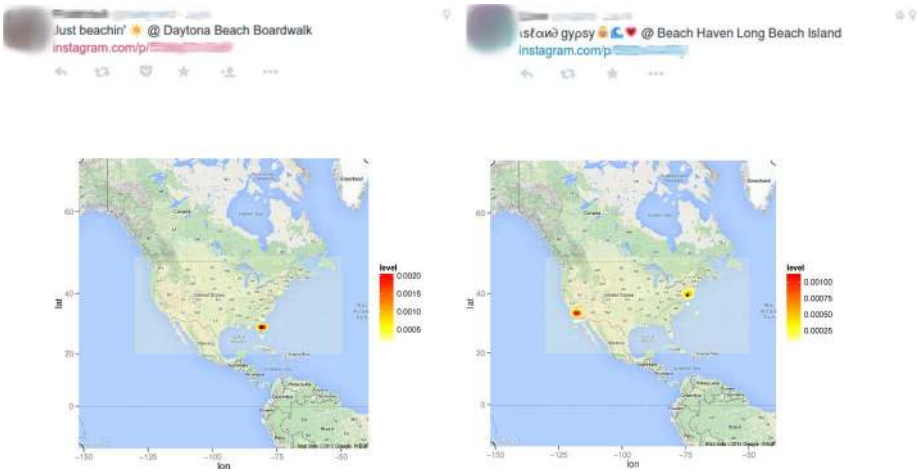


FIG 2. *Top: Two tweets with the keyword "beach". Bottom: Level sets of the estimated probability density of the tweet locations given the content of the tweets. The black dots indicate their true locations. See Sec. 3.3 for details.*

In Section 2, we describe our method in detail, and present connections with existing literature on Varying Coefficient methods and Spectral Series CDE. Section 3 presents several applications of FlexCode. Section 4 discusses convergence rates of the estimator, and Section 5 concludes the paper.

2. Methods

Assume we observe i.i.d. data $(\mathbf{X}_1, Z_1), \ldots, (\mathbf{X}_n, Z_n)$, where the covariates $\mathbf{x} \in \mathbb{R}^D$ with D potentially large, and the response $Z \in [0, 1]$.[1] Our goal is to estimate the full density $f(z|\mathbf{x})$ rather than, e.g., only the conditional mean $\mathbb{E}[Z|\mathbf{x}]$ and conditional variance $\mathbb{V}[Z|\mathbf{x}]$. We propose a novel "varying coefficient" series

[1] More generally, $\mathbf{x}$ can represent functional data, distributions, as well as mixed continuous and discrete data; see Sec. 3 for examples. The response z can also be multivariate (Sec. 2.2) or discrete [28, Sec. 4.2].

approach, where we start by specifying an orthonormal basis $(\phi_i)_{i\in\mathbb{N}}$ for $\mathcal{L}^2(\mathbb{R})$. This basis will be used to model the density $f(z|\mathbf{x})$ *as a function of* z. As we shall see, each coefficient in the expansion can be directly estimated via a regression. Note that there is a wide range of (orthogonal) bases one can choose from to capture any challenging shape of the density function of interest [37]. For instance, a natural choice for reasonably smooth functions $f(z|\mathbf{x})$ is the Fourier basis:

$$\phi_1(z) = 1; \quad \phi_{2i+1}(z) = \sqrt{2}\sin(2\pi i z),\ i \in \mathbb{N}; \quad \phi_{2i}(z) = \sqrt{2}\cos(2\pi i z),\ i \in \mathbb{N}$$

Alternatively, one can use wavelets or related bases to capture inhomogeneities in the density (see Sec. 3.4 for an example), and indicator functions to model discrete responses [28, Sec. 4.2].

Smoothing using orthogonal functions is per se not a new concept [9, 56]. The novelty in FlexCode is that we, by using an orthogonal series approach for the response variable, can convert a challenging high-dimensional conditional density estimation problem to a simpler high-dimensional regression (point estimation) problem.

For fixed $\mathbf{x} \in \mathbb{R}^D$, we write

$$f(z|\mathbf{x}) = \sum_{i\in\mathbb{N}} \beta_i(\mathbf{x})\phi_i(z). \tag{2.1}$$

Note that our model is fully nonparametric: Equation 2.1 hold as long as, for every $\mathbf{x}$, $f(z|\mathbf{x})$ is $\mathcal{L}^2(\mathbb{R})$ integrable as a function of z. Furthermore, because the $\{\phi_i\}_{i\in\mathbb{N}}$ basis functions are orthogonal to each other, the expansion coefficients are given by

$$\beta_i(\mathbf{x}) = \langle f(.|\mathbf{x}), \phi_i\rangle = \int_{\mathbb{R}} \phi_i(z) f(z|\mathbf{x}) dz = \mathbb{E}\left[\phi_i(Z)|\mathbf{x}\right]. \tag{2.2}$$

That is, each "varying coefficient" $\beta_i(\mathbf{x})$ in Eq. 2.1 is a *regression function*, or conditional expectation. This suggests that we, for fixed i, estimate $\beta_i(\mathbf{x})$ by regressing $\phi_i(z)$ on $\mathbf{x}$ using the sample $(\mathbf{X}_1, \phi_i(Z_1)), \ldots, (\mathbf{X}_n, \phi_i(Z_n))$.

We define our FlexCode estimator of $f(z|\mathbf{x})$ as

$$\widehat{f}(z|\mathbf{x}) = \sum_{i=1}^{I} \widehat{\beta}_i(\mathbf{x})\phi_i(z), \tag{2.3}$$

where the results from the regression,

$$\widehat{\beta}_i(\mathbf{x}) = \widehat{\mathbb{E}}\left[\phi_i(Z)|\mathbf{x}\right],$$

model how the density varies in covariate space. The cutoff I in the series expansion is a tuning parameter that controls the bias-variance tradeoff in the final density estimate. Generally speaking, the smoother the density, the smaller the value of I; see Sec. 4 Theory for details. In practice, we use cross-validation or data splitting (Sec. 2.1) to tune parameters.

With FlexCode, the problem of high-dimensional conditional density estimation boils down to choosing appropriate methods for estimating the regression functions $\mathbb{E}\left[\phi_i(Z)|\mathbf{x}\right]$, $i = 1, \ldots, I$. The key advantage of FlexCode is its flexibility: By taking advantage of new and existing regression methods, we can adapt to *different structures* in the data (e.g., manifolds, irrelevant covariates as well as different relationships between $\mathbf{x}$ and the response Z), and we can handle *different types of data* (e.g. mixed data, functional data, and so on). We will further explore this topic in Secs. 3–4.

2.1. Loss function and tuning of parameters

For a given estimator $\widehat{f}(z|\mathbf{x})$, we measure the discrepancy between $\widehat{f}(z|\mathbf{x})$ and $f(z|\mathbf{x})$ via the loss function

$$\begin{aligned} L(\widehat{f}, f) &= \iint \left(\widehat{f}(z|\mathbf{x}) - f(z|\mathbf{x})\right)^2 dP(\mathbf{x})dz \\ &= \iint \widehat{f}^2(z|\mathbf{x})dP(\mathbf{x})dz - 2\iint \widehat{f}(z|\mathbf{x})f(z,\mathbf{x})d\mathbf{x}dz + C, \end{aligned} \tag{2.4}$$

where C is a constant that does not depend on the estimator.

To tune the parameter I, we split the data into a training and a validation set. We use the training set to estimate each regression function $\beta_i(\mathbf{x})$. We then use the validation set $(z'_1, \mathbf{x}'_1), \ldots, (z'_{n'}, \mathbf{x}'_{n'})$ to estimate the loss (2.4) (up to the constant C) according to:

$$\widehat{L}(\widehat{f}, f) = \sum_{i=1}^{I} \frac{1}{n'} \sum_{k=1}^{n'} \widehat{\beta}_i^2(\mathbf{x}'_k) - 2\frac{1}{n'} \sum_{k=1}^{n'} \widehat{f}(z'_k|\mathbf{x}'_k), \tag{2.5}$$

This estimator is consistent because of the orthogonality of the basis $\{\phi_i\}_i$. We choose the tuning parameters with the smallest estimated loss $\widehat{L}(\widehat{f}, f)$. Algorithm 1 summarizes our procedure. In line 3, we split the training data in two parts to tune the parameters associated with the regression using the standard $\mathcal{L}^2(\mathbb{R})$ regression loss, i.e., $\mathbb{E}[(W - \widehat{\beta}_i(\mathbf{X}))^2]$.

In terms of computational efficiency, FlexCode is typically faster than existing methods for conditional density estimation (see Section 3), especially in high dimensions and for massive data sets. If the FlexCode estimator is based on a scalable regression procedure (e.g., Raykar [46], Desai et al. [7], Zhang et al. [58], Dai et al. [6]), then the resulting conditional density estimator will scale as well. Furthermore, FlexCode is naturally suited for parallel computing, as one can estimate each of the I regression functions separately and then combine the estimates according to Eq. (2.3). Our implementation of FlexCode is available at `https://github.com/rizbicki/FlexCoDE`, and implements a parallel version of the estimator. For the final density estimate (Step 9 in Algorithm 1), we apply the same techniques as in Izbicki and Lee [28, Section 2.2] to remove potentially negative values and spurious bumps.

Algorithm 1 FlexCode

Input: Training data; validation data; maximum cutoff I_0; orthonormal basis $\{\phi_i\}_i$; regression method and grid of tuning parameters for regression.

Output: Estimator $\widehat{f}(z|\mathbf{x})$

for all $i \leq I_0$ **do**
  Compute $\mathcal{D} = (\mathbf{X}_1, W_1), \ldots, (\mathbf{X}_n, W_n)$, where $W_k := \phi_i(Z_k)$
  Estimate the regression $\beta_i(\mathbf{x}) = \mathbb{E}[W|\mathbf{x}]$ using $\mathcal{D}$.
end for
for all $I \leq I_0$ **do**
  Calculate the estimated loss $\widehat{L}(\widehat{f}_I, f)$ on the validation set ▷ Eq. (2.5)
  ▷ $\widehat{f}_I$ is the estimator in Eq. (2.3)
end for
Define $\widehat{f}(z|\mathbf{x}) = \arg\min_{\widehat{f}_I} \widehat{L}(\widehat{f}_I, f)$
return $\widehat{f}(z|\mathbf{x})$

2.2. Extension to vector-valued responses

By tensor products, one can directly extend FlexCode to cases where the response variable $\mathbf{Z}$ is vector-valued. For instance, if $\mathbf{Z} \in \mathbb{R}^2$, consider the basis

$$\{\phi_{i,j}(\mathbf{z}) = \phi_i(z_1)\phi_j(z_2) : i, j \in \mathbb{N}\},$$

where $\mathbf{z} = (z_1, z_2)$, and $\{\phi_i(z_1)\}_i$ and $\{\phi_j(z_2)\}_j$ are bases for functions in $\mathfrak{L}^2(\mathbb{R})$. Then, let

$$f(\mathbf{z}|\mathbf{x}) = \sum_{i,j\in\mathbb{N}} \beta_{i,j}(\mathbf{x})\phi_{i,j}(\mathbf{z}),$$

where the expansion coefficients

$$\beta_{i,j}(\mathbf{x}) = \langle f(.|\mathbf{x}), \phi_{i,j}\rangle = \int_{\mathbb{R}^2} \phi_{i,j}(\mathbf{z}) f(\mathbf{z}|\mathbf{x}) d\mathbf{z} = \mathbb{E}[\phi_{i,j}(\mathbf{Z})|\mathbf{x}].$$

Note that each $\beta_i(\mathbf{x})$ is a regression function of a *scalar* response. In other words, the FlexCode framework allows one to estimate multivariate conditional densities by only using regression estimators of scalar responses.

Remark: To avoid tensor products, one can alternatively compute a *spectral basis* $\{\phi_i(\mathbf{z})\}_{i\geq 0}$ [36]. This basis is orthonormal with respect to the density $f(\mathbf{z})$ and adapts to the density's intrinsic geometry. The expansion coefficients are then given by $\beta_i(\mathbf{x}) = \langle f(.|\mathbf{x}), \phi_i\rangle_{f(\mathbf{z})} = \int_{\mathbb{R}^d} \phi_i(\mathbf{z}) f(\mathbf{z}|\mathbf{x}) f(\mathbf{z}) d\mathbf{z} = \mathbb{E}[\phi_i(\mathbf{Z}) f(\mathbf{Z})|\mathbf{x}]$, in which case one needs to estimate $f(\mathbf{Z})$ as well.

2.3. Connection to other methods

Varying-Coefficient Models. The model $f(z|\mathbf{x}) = \sum_{i\in\mathbb{N}} \beta_i(\mathbf{x})\phi_i(z)$ can be viewed as a *fully nonparametric* varying-coefficient model. *Varying-coefficient models* [21] are often seen as semi-parametric models or as extensions of classical linear models, in which a function η is modeled as $\eta = \sum_{i=1}^{d} \beta_i(\mathbf{x})u_i$, where $\beta_i(\mathbf{x})$

are smooth functions of the predictors $\mathbf{x}$, and $u_1, \ldots, u_d$ are other predictors. In our case, we have a fully nonparametric model, because $d \longrightarrow \infty$ and $(u_i)_{i\geq 1} := \{\phi_i(z)\}_{i\geq 1}$ is a basis of $\mathcal{L}^2(\mathbb{R})$.

Spectral Series CDE. FlexCode recovers the spectral series conditional density estimator of Izbicki and Lee [28] if each $\beta_i(\mathbf{x})$ is estimated via a *spectral series regression* [36]. Indeed, let $\{\psi_j\}_j$ be a spectral basis for $\mathbf{x}$, where by construction $\int_{\mathcal{X}} \psi_i(\mathbf{x})\psi_j(\mathbf{x}) dP(\mathbf{x}) = \delta_{i,j} \stackrel{\text{def}}{=} \mathbb{I}(i=j)$ [28, Sec. 2]. In spectral series CDE, one writes the conditional density as $f(z|\mathbf{x}) = \sum_{i\geq 1}\sum_{j\geq 1} \beta_{i,j}\phi_i(z)\psi_j(\mathbf{x})$, where the coefficients

$$\beta_{i,j} = \iint f(z|\mathbf{x})\phi_i(z)\psi_j(\mathbf{x})\, dP(\mathbf{x})dz = \mathbb{E}\left[\phi_i(Z)\psi_j(\mathbf{X})\right]. \tag{2.6}$$

Now, a spectral series regression for $\beta_i(\mathbf{x}) = \mathbb{E}\left[\phi_i(Z)|\mathbf{x}\right]$ is based on the model $\beta_i(\mathbf{x}) = \sum_{j\geq 1}\gamma_j^{(i)}\psi_j(\mathbf{x})$, where

$$\begin{aligned}\gamma_j^{(i)} &= \int \beta_i(\mathbf{x})\psi_j(\mathbf{x})\, dP(\mathbf{x}) = \int \mathbb{E}\left[\phi_i(Z)|\mathbf{x}\right]\psi_j(\mathbf{x})\, dP(\mathbf{x}) = \\ &= \int \mathbb{E}\left[\phi_i(Z)\psi_j(\mathbf{X})|\mathbf{x}\right]\, dP(\mathbf{x}) = \mathbb{E}\left[\phi_i(Z)\psi_j(\mathbf{X})\right].\end{aligned} \tag{2.7}$$

By inserting $\beta_i(\mathbf{x})$ into Eq. 2.1, we see that Spectral Series CDE [28] is a special case of FlexCode. Henceforth, we will refer to this version of FlexCode as *FlexCode-Spec*.

Remark: Using similar arguments, one can show that FlexCode recovers the orthogonal series CDE of Efromovich [9] if each $\beta_i(\mathbf{x})$ is estimated via traditional orthogonal series regression. However, as discussed in Izbicki and Lee [28], traditional series approaches via tensor products quickly become intractable in high dimensions. Nevertheless, it is interesting to note that FlexCode forms a very large family of CDE approaches that includes Spectral Series CDE and traditional orthogonal series CDE as special cases.

3. Experiments

In what follows, we compare the following estimators:

- FlexCode is our proposed series approach. We implement six versions of FlexCode, where we use different regression methods to compute the coefficients $\widehat{\beta}_i(\mathbf{x}) = \widehat{\mathbb{E}}\left[\phi_i(Z)|\mathbf{x}\right]$ in Eq. 2.3. *FlexCode-SAM* is based on Sparse Additive Models [45].[2] *FlexCode-NN* is based on Nearest Neighbors regression [22]. *FlexCode-Spec* uses Spectral Series regression [36] and is, as shown in Sec. 2.3, the same as Spectral Series CDE, the conditional density estimator in Izbicki and Lee [28]. For mixed data types, we implement *FlexCode-RF*,

[2]Sparse additive regression models can be useful even if the true coefficients $\beta_i(\mathbf{x})$ are not additive, because of the curse of dimensionality and the ability of sparse additive models to identify irrelevant coefficients without too restrictive assumptions.

which estimates the regression functions via random forests [5], and for functional data, we use *FlexCode-fKR*, where the coefficients in the model are estimated via functional kernel regression [16]. Finally, in Sec. 3.4, we illustrate how *FlexCode-SDM* can extend Support Distribution Machines (SDM; Sutherland et al. [52]) and other distribution regression methods to estimating conditional densities on *sample sets* or groups of vectors.

- *KDE* is the kernel density estimator $\widehat{f}(z|\mathbf{x}) := \widehat{f}(z,\mathbf{x})/\widehat{f}(\mathbf{x})$, where $\widehat{f}(z,\mathbf{x})$ and $\widehat{f}(\mathbf{x})$ are standard multivariate kernel density estimators. We rescale the data to have the same mean and variance in each direction, and we assume an isotropic Gaussian kernel for both $\mathbf{x}$ and z, i.e.,

$$\widehat{f}(z|\mathbf{x}) = \frac{\sum_{i=1}^{n} K_{h_x}(\|\mathbf{x}-\mathbf{X}_i\|)K_{h_z}(z-Z_i)}{\sum_{i=1}^{n} K_{h_x}(\|\mathbf{x}-\mathbf{X}_i\|)},$$

  where $K_h(t) = h^{-d}K(t/h)$ denotes an isotropic Gaussian kernel with bandwidth h in d dimensions.
- KDE_{Tree} is the multivariate kernel density estimator $\widehat{f}(z|\mathbf{x}) := \widehat{f}(z,\mathbf{x})/\widehat{f}(\mathbf{x})$, where the estimators $\widehat{f}(z,\mathbf{x})$ and $\widehat{f}(\mathbf{x})$ have a different bandwidth for each component of $\mathbf{x}$ [20]; i.e.,

$$\widehat{f}(z|\mathbf{x}) = \frac{\sum_{i=1}^{n} K_h(\mathbf{x}-\mathbf{X}_i)K_{h_z}(z-Z_i)}{\sum_{i=1}^{n} K_h(\mathbf{x}-\mathbf{X}_i)},$$

  where $K_h(\mathbf{x}-\mathbf{X}_i) = (h_1\dots h_d)^{-1}\prod_{j=1}^{d} K\left(\frac{x_j-X_{ij}}{h_j}\right)$ for data $\mathbf{X}_i = (X_{i1},\dots,X_{id})$ and a bandwidth vector $h = (h_1,\dots,h_d)$. We use the R package `np` [23] with kd-trees and Epanechnikov kernels for computational efficiency [19, 25].
- *kNN* is a kernel nearest neighbors approach [59, 30] to conditional density estimation; it is defined as

$$\widehat{f}(z|\mathbf{x}) \propto \sum_{j\in\mathcal{N}_k(\mathbf{x})} K_\epsilon\left(z-Z_j\right),$$

  where $\mathcal{N}_k(\mathbf{x})$ is the set of the k closest neighbors to $\mathbf{x}$ in the training set, and K_ϵ is a multivariate (isotropic) Gaussian kernel with bandwidth ϵ.
- *fkDE* is a nonparametric conditional density estimator for functional data [43]. It is defined as

$$\widehat{f}(z|\mathbf{x}) = \frac{\frac{1}{h_z}\sum_{i=1}^{n} K\left(\frac{d(x,X_i)}{h_x}\right)K_0\left(\frac{z-Z_i}{h_z}\right)}{\sum_{i=1}^{n} K\left(\frac{d(x,X_i)}{h_x}\right)},$$

  where d is a distance measure in the (functional) space of the data, K and K_0 are isotropic kernel functions, and h_x and h_z are tuning parameters.

Note that for *regression*, SAM is designed to work well when there is a small number of relevant covariates, and both Spectral Series Regression and Nearest Neighbors Regression perform well when the covariates exhibit a low intrinsic

dimensionality. To our knowledge, KDE_{Tree} is the only CDE method that can handle mixed data types.

3.1. Toy examples

By simulation, we create toy versions of common scenarios with different structures in data and different types of data. We use 700 data points for training, 150 for validation and 150 for testing the methods. Each simulation is repeated 200 times.

Different structures in data

- **Irrelevant Covariates.** In this example, we generate data according to $Z|\mathbf{x} \sim N(x_1, 0.5)$, where $\mathbf{X} = (X_1, \ldots, X_d) \sim N(\mathbf{0}, I_d)$, that is, only the first covariate influences the response.
- **Data on Manifold.** Here we let $Z|\mathbf{x} \sim N(\theta(\mathbf{x}), 0.5)$, where $\mathbf{x} = (x_1, \ldots, x_d)$ lie on a unit circle embedded in a D-dimensional space, and $\theta(\mathbf{x})$ is the angle corresponding to the position of $\mathbf{x}$. For simplicity, we assume that the data are uniformly distributed on the manifold; i.e., we let $\theta(\mathbf{x}) \sim Unif(0, 2\pi)$.
- **Non-Sparse Data.** Finally, we consider data with no sparse (low-dimensional) structure. We assume $Z|\mathbf{x} \sim N(\overline{\mathbf{x}}, 0.5)$, where $\mathbf{X} = (X_1, \ldots, X_d) \sim N(\mathbf{0}, I_d)$.

Different types of data

- **Mixed Data Types.** Few existing CDE methods can handle mixed data types; the only other method the authors are aware of is KDE_{Tree}. For our study, we generate mixed categorical and continuous data, where the categorical covariates $(X_1, \ldots, X_{D/2})$ are i.i.d. $Unif\{c_1, c_2, c_3, c_4, c_5\}$, and the continuous covariates $(X_{D/2+1}, \ldots, X_D)$ are i.i.d. $N(0, 1)$. The response is given by

$$Z|\mathbf{x} \sim \begin{cases} N(x_{D/2+1}, 0.5) & \text{if } x_1 \in \{c_1, c_2\} \\ 10 + 2N(x_{D/2+2}, 0.5) & \text{if } x_1 \in \{c_3, c_4, c_5\} \end{cases}$$

- **Functional Data.** We also consider spectrometric data for finely chopped pieces of meat. These high-resolution spectra are available[3] as a benchmark for functional regression models (see, e.g., Ferraty et al. [17]), where the task is to predict the fat content of a meat sample on the basis of its near infrared absorbance spectrum. In our study, we use 215 samples to estimate conditional densities. The covariates are spectra of light absorbance as functions of the wavelength, and the response is the fat content of a piece

[3] http://lib.stat.cmu.edu/datasets/tecator; the original data source is Tecator AB.

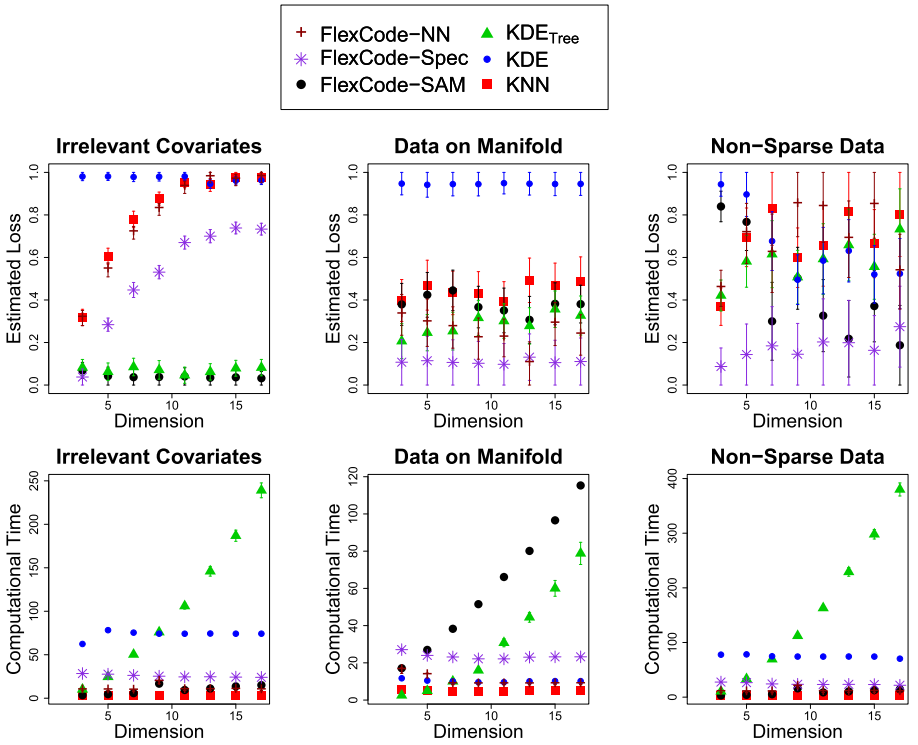


FIG 3. *Examples with different structures in data.* Top row: *Estimated loss as a function of the dimension D.* Bottom row: *Computational time as a function of D. With a properly chosen regression method, FlexCode performs better than the other estimators (KDE$_{Tree}$, KDE, and kNN).*

of meat. We compare the functional kernel density estimator (*fKDE*) with a FlexCode approach (*FlexCode-fKR*), where the coefficients in the model are estimated via functional kernel regression [16]. We follow Ferraty et al. [17] and implement both methods with the kernel function $K(u) = 1 - u^2$ and the $\mathcal{L}^2(\mathbb{R})$ norm between the second derivatives of the spectra as a distance measure. We use 70% of the data points for training, 15% for validation and 15% for testing; the experiment is repeated 100 times by randomly splitting the data.

3.1.1. Results

Figures 3–4 show the results for the toy data. Our main observations are:

- **Irrelevant Covariates.** In terms of estimated loss (Fig. 3, top left), both *FlexCode-SAM* and *KDE$_{Tree}$* outperform the other methods. However, in terms of computational time (Fig. 3, bottom left), *FlexCode-SAM*

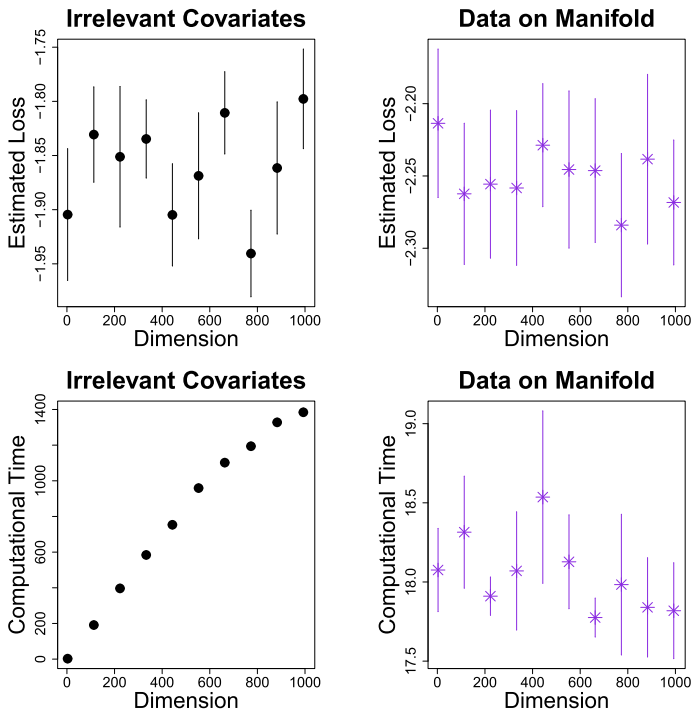


FIG 4. *Different structures in data for large values of D. Estimated loss (*top row*) and computational time (*bottom row*) of FlexCode in the two settings "Irrelevant Covariates" (*left*; implemented with FlexCode-SAM) and "Data on Manifold" (*right*; implemented with FlexCode-Spec). These two estimators yield the best results in Fig. 3; here we see their behavior in higher dimensions.*

is clearly faster than KDE_{Tree} as the dimension D of the data grows. When $D = 17$, each fit with KDE_{Tree} already takes an average of 240 seconds (4 minutes) on an Intel i7-4800MQ CPU 2.70GHz processor, compared to 22 seconds for *FlexCode-SAM*. Fig. 4, left, shows that the loss of *FlexCode-SAM* remains the same even for large $D \sim 1000$, although fitting the estimator becomes computationally more challenging in high dimensions. Nevertheless, fitting KDE_{Tree} would be unfeasible for $D > 50$.

- **Data on Manifold.** *FlexCode-Spec* has the best statistical performance, followed by *FlexCode-NN* and KDE_{Tree} (Fig. 3, top center). As before, the computational time of KDE_{Tree} increases rapidly with the dimension (Fig. 3, center bottom). For these data, *FlexCode-SAM* is slow as well even for moderate D, perhaps because SAM cannot find sparse representations of the regression functions. On the other hand (see Fig. 4, *right*), *FlexCode-Spec* has a computational time that is almost constant as a function of D and the statistical performance remains the same even for large D. The latter result is consistent with our previous findings that spectral series adapt to the intrinsic dimension of the data [29, 28, 36].

- **Non-Sparse Data.** For this example, *FlexCode-Spec* and *FlexCode-SAM* are the best estimators.
- **Mixed Data Types.** *FlexCode-RF* yields better results than KDE_{Tree} (its only competitor in this setting) both in terms of estimated loss and computational time; see Fig. 5. The computational advantage is especially obvious for larger values of D. When the dimension $D = 56$, each fit of KDE_{Tree} takes an average of 2250 seconds ($\approx$ 37 minutes) on an Intel i7-4800MQ CPU 2.70GHz processor, compared to 304 seconds ($\approx$ 5 minutes) for *FlexCoDR-RF*.
- **Functional Data.** FlexCode via Functional kernel regression improves upon the results of the traditional Functional kernel density estimator with an estimated loss of -2.78 (0.07) instead of -2.08 (0.03).

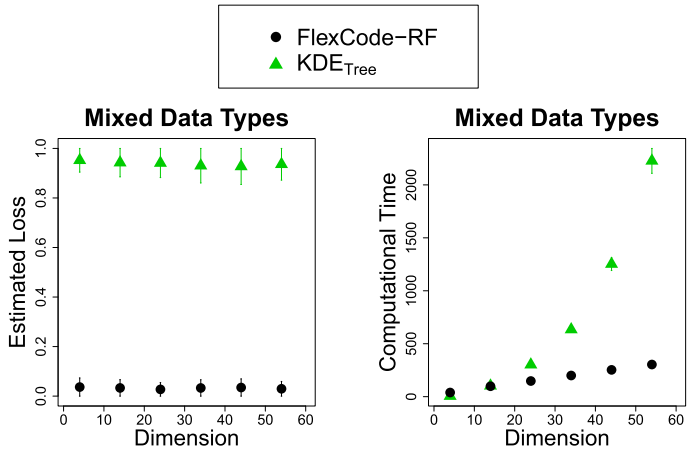


FIG 5. *Example with mixed data. Estimated loss (*left*) and computational time (*right*) for FlexCode via Random Forests (FlexCode-RF) and* KDE_{Tree}*. Few conditional estimators can handle covariates with mixed data types, but FlexCode is flexible enough to adapt to this setting.*

3.2. Photometric redshift estimation

Our first application is photometric redshift estimation. Redshift (a proxy for a galaxy's distance from the Earth) is a key quantity for inferring cosmological model parameters. Redshift can be estimated with high precision via spectroscopy but the resource considerations of large-scale sky surveys call for *photometry* – a much faster measuring technique, where the radiation from an astronomical objects is generally coarsely recorded via $\sim$5–10 broad-band filters. In photometric redshift estimation, the goal is to estimate the redshift z of a galaxy based on its observed photometric covariates $\mathbf{x}$, using a sample of galaxies with spectroscopically confirmed redshifts. Because of degeneracies (two galaxies

with different redshifts can have similar photometric signatures) and because of complicated observational noise, probability densities of the form $f(z|\mathbf{x})$ better describe the relationship between $\mathbf{x}$ and z than the regression $\mathbb{E}(z|\mathbf{x})$ does.

In this example, we test our CDE methods on $n = 752$ galaxies from *COSMOS*, with $D = 37$ covariates derived from a variety of photometric bands (these data were obtained from T. Dahlen 2013, private communication; see Izbicki et al. [30] for additional details). Figure 6 summarizes the results. All versions of FlexCode improve upon the traditional estimators. The best performance is achieved for FlexCode via Sparse Additive Models (*FlexCode-SAM*), which indicates that only a subset of the 37 covariates are relevant for redshift estimation; for these data, *FlexCode-SAM* selected ≈ 18 variables in each regression, and three out of the 37 covariates were present in more than 75% of the regressions.

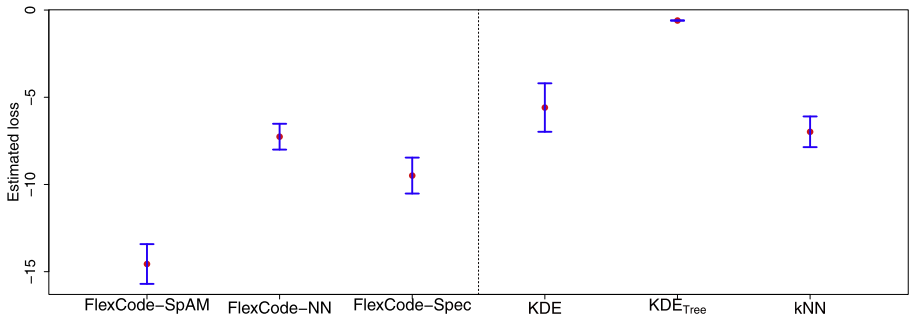


FIG 6. *Estimated losses of conditional density estimators for photometric redshift prediction. All versions of FlexCode (to the left) improve upon the traditional estimators (to the right)*

3.3. Twitter data

Twitter is a social network where each user is able to post a small text (a tweet) containing at most 140 characters. Information about the location of the post is available upon user permission, but only a few users allow this information to be publicly shared. Here we use samples with known locations to estimate the location of tweets where this information has not been shared publicly.

Note that most literature on the topic concerns creating *point estimates* for locations (see, e.g., Rodrigues et al. [47] and references therein). In this work, we estimate the *full conditional distribution* of latitude and longitude given the content of the tweet; that is, we estimate $f(\mathbf{z}|\mathbf{x})$, where $\mathbf{x}$ are covariates extracted from the tweets and $\mathbf{z} = (z_1, z_2)$ is the pair latitude/longitude.

Our data set contains ≈8000 tweets in the USA from July 2015 with the word "beach". We extract 500 covariates via a bag-of-words method with the most frequent unigrams and bigrams [38]. As we only expect a few of the 500 covariates to be relevant to locating the tweets, we implement FlexCode via sparse

additive models. Figure 2 shows two examples of estimated densities; see Supplementary material for additional examples. To our knowledge, no other fully nonparametric conditional density estimation method can be directly applied to these types of data where there are many irrelevant variables.

Moreover, because *FlexCode-SAM* is based on sparse additive models, we can find out which covariates are most relevant for predicting location. For the example in Fig. 2, left, the expressions "beachin","boardwalk", and "daytona" are included in at least 33% of the estimated regression functions. For the example to the right, the relevant covariates are "long beach", "island", "long", and "haven".

3.4. From distribution regression to "Distribution CDE": Estimating the mass of a galaxy cluster from sample sets of galaxy velocities

Distribution regression and classification is a recent emerging field of machine learning. Instead of treating individual data points (or feature vectors) as covariates, these methods operate on *sample sets*, where each set is a sample from some underlying feature distribution; see Sutherland et al. [52] and references within. Here we show that FlexCode extends to sample sets as well; our application is estimation of the mass of a galaxy cluster given the line-of-sight velocities of the galaxies in the cluster.

Galaxy clusters, the most massive gravitationally bound systems in the Universe, can contain up to ∼1000 galaxies. These structures are a rich source of information on astrophysical processes and cosmological parameters, but to use galaxy clusters as cosmological probes one needs to accurately measure their masses. A standard approach is to employ the classical virial theorem and directly relate the mass of a cluster to the line-of-sight (LOS) galaxy velocity dispersion, i.e., the variance of the measured galaxy velocities in the cluster [11]. Recently, Ntampaka et al. [40] and Ntampaka et al. [41] have shown that one can significantly improve such mass predictions by taking advantage of the entire LOS *velocity distribution* of galaxies instead of only the dispersion (i.e., a summary of the distribution). Here we show that FlexCode can further improve these results.

The general set-up is that we observe data of the form $(\mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_1^{(J_1)}, z_1), \ldots, (\mathbf{x}_I^{(1)}, \ldots, \mathbf{x}_I^{(J_I)}, z_I)$, where z_i is the mass of the i-th cluster for $i = 1, \ldots, I$; and $\mathbf{x}_i^{(j)}$ is a vector of galaxy observables (such as LOS velocity and the projected distance from the cluster center) for the j-th galaxy in the i-th cluster. Note that different clusters i contain different numbers J_i of galaxies. The key idea behind Support Distribution Machines (SDMs; proposed for this application by Ntampaka et al. [40]) as well as other "distribution regression" methods [52], is to treat each sequence $\mathbf{x}_i^{(1)}, \ldots, \mathbf{x}_i^{(J_i)}$ as a sample from a probability distribution p_i, and to construct an appropriate kernel matrix on these sample sets. The task is then to predict a scalar (z_i) from a distribution (p_i) by estimating $\mathbb{E}[Z|p]$. Here we show how FlexCode extends regression on distributions to *conditional den-*

sity estimation on distributions; i.e., instead of providing a point estimate (and standard error) of the mass of a galaxy cluster, we estimate the full probability density $f(z|p)$ of the unknown mass of a galaxy cluster given galaxy observables. In our application, the response z_i is the logarithm of the cluster mass (log M) and the observables $\{x_i^j\}_{j=1}^{J_i}$ are scalar quantities that represent the absolute values of galaxy velocities along *one* line-of-sight.

Like Ntampaka et al. [40], we use the Kullback-Leibler (KL) divergence to measure similarity between pairs of velocity distributions, and we estimate the divergence from the observed galaxy velocities with the estimator from Wang et al. [55]. The details are as follows: Let p_A and p_B denote velocity distributions for clusters A and B, respectively. Define the kernel $k(p_A, p_B) = \exp(-\mathrm{KL}(p_A, p_B)/\sigma^2)$, where $\mathrm{KL}(p_A, p_B)$ is the Kullback-Leibler divergence between p_A and p_B. We estimate the KL divergence via Wang et al's k nearest neighbors method for $k = 2$. That is, let X_A denote the set of LOS velocities associated with the n galaxies of cluster A, and let X_B denote the set of velocities associated with the m galaxies of cluster B. The estimated KL divergence from p_A to p_B is given by

$$\mathrm{KL}_{n,m}(X_A, X_B) = \frac{d}{n}\sum_{i=1}^{n} \log \frac{\nu_k(i)}{\rho_k(i)} + \log \frac{m}{n-1},$$

where $\nu_k(i)$ is the Euclidean distance from the covariates (in this case, the LOS velocity) of the i-th galaxy in X_A to its k-th nearest neighbor in X_B, $\rho_k(i)$ is the Euclidean distance from the covariates (the LOS velocity) of the i-th galaxy in X_A to its k-th nearest neighbor in X_A, and d is the number of galaxy observables (in this example, $d = 1$). As the computed kernel matrix $k(X_A, X_B) = \exp(-\mathrm{KL}_{n,m}(X_A, X_B)/\sigma^2)$ may not be positive semi-definite (PSD), we project the matrix to the closest PSD matrix in Frobenius norm [24].

Using the PSD kernel matrix, we then estimate the conditional density $f(z|p)$. We compare four approaches to conditional density estimation on distributions, which as in the rest of the paper use a Fourier basis in z;

- Functional KDE: the functional kernel density estimator [43],
- FlexCode-NN: FlexCode with Nearest Neighbors regression,
- FlexCode-Spec: FlexCode with Spectral Series regression,
- FlexCode-SDM: FlexCode with SDM regression.

In the experiments, we also include a FlexCode estimator that use a wavelet basis in z;

- FlexCode$_W$-SDM: FlexCode with SDM regression in x, and Daubechies wavelets with 3 vanishing moments in z.

Our data consist of simulations of $n = 5028$ unique galaxy clusters with minimum mass of $1 \times 10^{14}\ M_\odot h^{-1}$; see Ntampaka et al. [40] for details. All four methods above are based on the same distance computation $\mathrm{KL}_{n,m}(X_A, X_B)$ with $k = 2$, and we use data splitting and the loss (2.5) for selecting tuning

parameters. For simplicity, we only consider one LOS for each cluster (the x-axis LOS in the catalog).

It is clear from Table 1 that the FlexCode-SDM and FlexCode$_W$-SDM estimates of conditional density are more accurate than the results from any other method. The coverage plots (see Appendix A for the definition) in the bottom panel of Fig. 7 also verify that these density estimates fit the observed data well.

TABLE 1
Estimated losses of conditional density estimates of galaxy cluster mass

Functional KDE	*FlexCode-NN*	*FlexCode-Spec*	*FlexCode-SDM*	*FlexCode$_W$-SDM*
-0.98 (0.02)	-1.60 (0.05)	-1.86 (0.04)	**-2.46 (0.09)**	**-2.71 (0.09)**

The top left panel of Figure 7 shows examples of density estimates from FlexCode$_W$-SDM for 16 randomly chosen clusters. Several of these distributions are bimodal, in which case regression estimates are not very informative. This can be further illustrated by Fig. 8. The left panel shows a scatter plot of the observed log masses versus the estimated conditional mean $\widehat{\mathbb{E}}[Z|p] := \int z\widehat{f}(z|p)dz$ for unimodal versus multimodal cases. The right panel shows a boxplot of the absolute fractional mass error $|\varepsilon|$ for the two populations; the fractional mass error ε is defined as [40]

$$\varepsilon = \frac{M_{\text{pred}} - M}{M},$$

where M is the observed cluster mass and M_{pred} is the predicted cluster mass. Much of the scatter can indeed be attributed to multimodal densities and non-standard prediction settings.

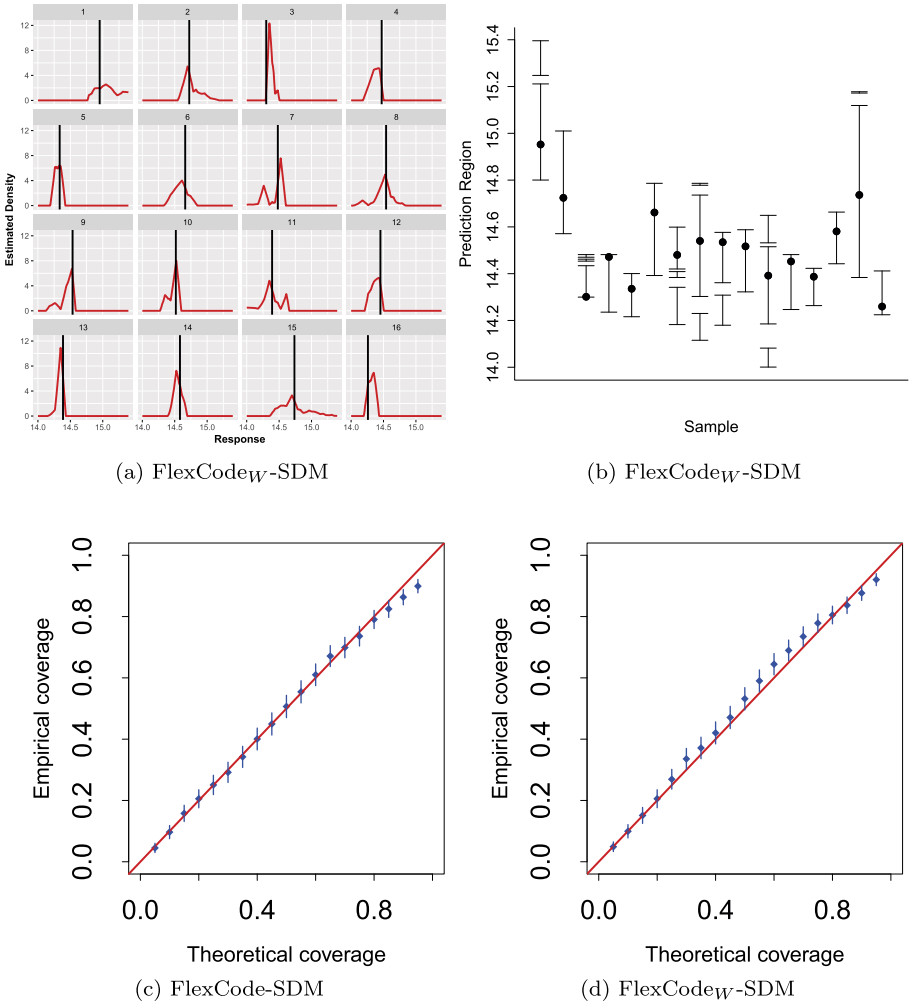
Finally, we notice that both the mean and the mode of FlexCode-SDM as well as FlexCode$_W$-SDM densities improve upon plain SDM regression. Table 2 compares the fractional mass error distributions of the predictions. By taking the mode of the FlexCode density we reduce the ε 68% scatter[4] from $\Delta\varepsilon \approx 0.24$ for standard SDM down to a width of ≈ 0.15 for FlexCode-SDM and of ≈ 0.17 for FlexCode$_W$-SDM with a mode estimator.

TABLE 2
Performance of different methods

	Mean fractional error	Median fractional error	68% scatter fractional error
SDM Regression	0.052	-0.004	0.244
FlexCode-SDM Mean	0.012	-0.036	-0.228
FlexCode-SDM Mode	0.003	-0.025	**0.152**
FlexCode$_W$-SDM Mean	-0.003	-0.046	0.210
FlexCode$_W$-SDM Mode	$5.6 * 10^{-5}$	-0.019	**0.168**

To summarize: FlexCode extends SDM to conditional density estimation on distributions, and the estimated densities produce better point estimates of cluster masses. The real advantage with FlexCode, however, is that we can more accurately quantify the uncertainty in the predictions and potentially improve

[4]The ε 68% scatter, $\Delta\varepsilon$, is the 68% quantile of the distribution of $|\varepsilon|$.

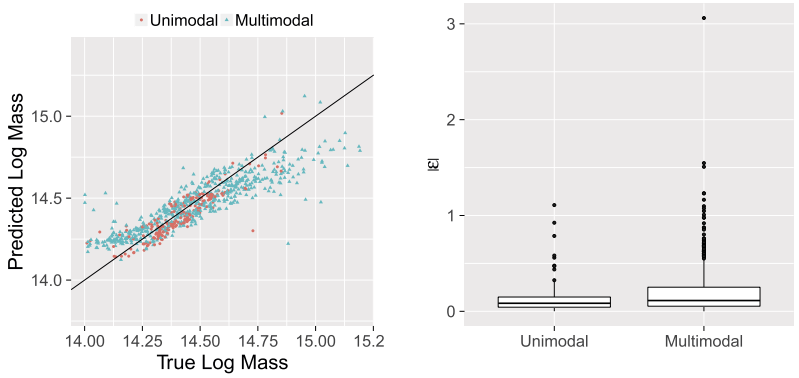


(a) FlexCode$_W$-SDM (b) FlexCode$_W$-SDM

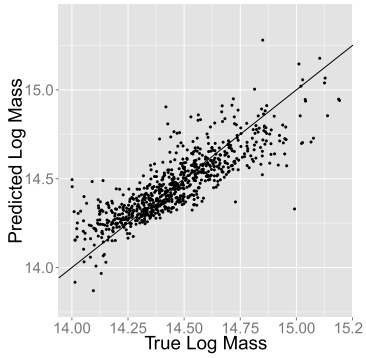
(c) FlexCode-SDM (d) FlexCode$_W$-SDM

FIG 7. Top left*: Estimated probability distributions of the log mass ("Response") of 16 randomly chosen clusters; the vertical lines show the true values, and the red curves are computed using FlexCode$_W$-SDM. Many of these densities are multimodal and asymmetric, indicating that standard prediction approaches may not accurately model the uncertainty in the mass estimates.* Top right*: 95% highest predictive density (HPD) regions for the same 16 clusters derived from the FlexCode$_W$-SDM estimates; the dots show the true values.* Bottom*: Coverage plots of the density estimates from FlexCode-SDM and FlexCode$_W$-SDM for the entire mock cluster catalog. The plots show that these density estimates fit the observed data well.*

inference for outliers or cases that are not well described by one-number summaries. For example, we can use the estimated densities to construct more informative highest predictive density (HPD) regions of the cluster mass, i.e., regions of the form $\{z : \widehat{f}(z|\mathbf{x}) \geq K\}$, where K is chosen in such a way that the regions have the desired coverage level (e.g., 95%). The top panels of Figure 7 shows



(a) FlexCode$_W$-SDM (b) FlexCode$_W$-SDM



(c) SDM Regression

FIG 8. Left: *Scatter plot of the predicted versus the true log masses for FlexCode$_W$-SDM when taking the conditional mean* $\widehat{\mathbb{E}}[Z|p] := \int z\widehat{f}(z|p)dz$ *("FlexCode$_W$-SDM Mean"). The red and blue dots denote clusters with unimodal and multimodal mass densities, respectively.* Center*: Boxplot of the the absolute fractional mass error* $|\varepsilon|$ *for the two populations. These results again indicate that much of the scatter in the mass error estimates are due to multimodal densities.* Right*: For comparison, we include a scatterplot of the predicted versus the true log mass masses for SDM regression as in [40], where we cannot extract this information.*

some examples of multimodal densities and their 95% HPD regions. In many cases, returning *a predictive region for the cluster mass* is a better alternative to just taking the mean or mode of the density. The coverage plot in the bottom right panel also indicates that the empirical coverage of these regions is indeed close to 95%.

4. Theory

In this section, we derive bounds and rates for FlexCode; that is, the conditional density estimator in Eq. 2.3. We use the notation $\widehat{f}_I(z|\mathbf{x})$ to indicate its dependence on the cutoff I.

We assume that f belongs to a set of functions which are not too "wiggly". For every $s > \frac{1}{2}$ and $0 < c < \infty$, let $W_\phi(s,c) = \{f = \sum_{i\geq 1} \theta_i \phi_i : \sum_{i\geq 1} a_i^2 \theta_i^2 \leq c^2\}$, where $a_i \sim (\pi i)^s$, denote the Sobolev space. For the Fourier basis $\{\phi_i\}_i$, this is the standard definition of Sobolev space [56]; it is the space of functions that have their s-th weak derivative bounded by c^2 and integrable in $\mathcal{L}^2(\mathbb{R})$. We enforce smoothness in the z-direction by requiring $f(z|\mathbf{x})$ to be in a Sobolev space for all $\mathbf{x}$. This is formally stated as Assumption 1, where β and C are used to link the Sobolev spaces at different x.

Assumption 1 (Smoothness in z direction)**.** $\forall \mathbf{x} \in \mathcal{X}$, $f(z|\mathbf{x}) \in W_\phi(s_\mathbf{x}, c_\mathbf{x})$, *where* $f(z|\mathbf{x})$ *is viewed as a function of* z, *and* $s_\mathbf{x}$ *and* $c_\mathbf{x}$ *are such that* $\inf_\mathbf{x} s_\mathbf{x} \stackrel{def}{=} \beta > \frac{1}{2}$ *and* $\int_\mathcal{X} c_\mathbf{x}^2 d\mathbf{x} \stackrel{def}{=} C < \infty$.

We also assume that each function $\beta_i(\mathbf{x})$ is estimated using a regression method with convergence rate $O(n^{-2\alpha/(2\alpha+d)})$, where typically α is a parameter related to the smoothness of the $\beta_i(\mathbf{x})$ function, and d is either the number of relevant covariates or the intrinsic dimension of $\mathbf{x}$. In other words, we assume that each regression *adapts* to sparse structure in the data. This is formally stated as Assumption 2.

Assumption 2 (Regression convergence)**.** *For every* $i \in \mathbb{N}$, *there exists some* $d \in \mathbb{N}$ *and* $\alpha > 0$ *such that*

$$\mathbb{E}\left[\int \left(\widehat{\beta}_i(\mathbf{x}) - \beta_i(\mathbf{x})\right)^2 d\mathbf{x}\right] = O(n^{-2\alpha/(2\alpha+d)})$$

Note that the smoothness parameter α must be the same for every $i \in \mathbb{N}$. Typically this assumption will hold because in many applications it is reasonable to assume that (i) if $\mathbf{x}_1$ is close to $\mathbf{x}_2$, then $f(z|\mathbf{x}_1)$ is also close to $f(z|\mathbf{x}_2)$ for every $z \in \mathbb{R}$ (in other words, $f(z|\mathbf{x})$ is smooth as a function of $\mathbf{x}$), and (ii) there is some structure in x (e.g., low intrinsic dimensionality) or in the relationship between x and z (e.g., sparsity), which the regression method for estimating β_i takes advantage of. Here are some examples where Assumption 2 holds:

(E1) $\widehat{\beta}_i$ is the k-nearest neighbors estimator [33], d is the intrinsic dimension of the covariate space and, for every $z \in [0,1]$, $f(z|\mathbf{x})$ is L-Lipschitz in $\mathbf{x}$ (in this case, $\alpha = 1$);

(E2) $\widehat{\beta}_i$ is a local polynomial regression [4], d is the intrinsic dimension of the covariate space and, for every $z \in [0,1]$, $f(z|\mathbf{x})$ is α times differentiable with all partial derivatives up to order α in $\mathbf{x}$ are bounded;

(E3) $\widehat{\beta}_i$ is the Rodeo estimator [35], d is the number of variables that affect the distribution of Z and, for every z, all partial derivatives of $f(z|\mathbf{x})$ up to fourth order in $\mathbf{x}$ are bounded (in this case, $\alpha = 2$);

(E4) $\widehat{\beta}_i$ is the regression estimator from Bertin and Lecué [2], d is the number of variables that affect the distribution of Z,[5] and, for every $z \in [0,1]$, $f(z|\mathbf{x})$ is α-Hölderian in $\mathbf{x}$;

[5]That is, there exists a subset $R \subseteq \{1,\ldots,D\}$ with $|R| = d$ such that $f(z|\mathbf{x}) = f(z|(x_i)_{i\in R})$.

(E5) $\widehat{\beta}_i$ is the Spectral series regression [36], d is the intrinsic dimension of the covariate space and, for every $z \in [0,1]$, $f(z|\mathbf{x})$ is smooth with respect to P_X according to $\int ||\nabla f(z|\mathbf{x})||^2 dS(\mathbf{x}) < \infty$ for a smoothed version $S(\mathbf{x})$ of f (in which case $\alpha = 1$);

(E6) $\widehat{\beta}_i$ is a local linear functional regression [1], the predictor X is a function talking values in $\mathcal{L}^2([0,1])$, X is fractal of order τ, and, for every $z \in [0,1]$, $f(z|\mathbf{x})$ is twice differentiable with a continuous second derivative (yielding rates with $\alpha = 2$ and $d = \tau$.)

In essence, Assumption 2 holds for examples E1–E6 because smoothness in $f(z|\mathbf{x})$ (seen as a function of $\mathbf{x}$) implies smoothness of the $\beta_i(\mathbf{x})$ functions in FlexCode. We refer to Appendix A1 for details and proofs. (See also, e.g., Yang and Tokdar [57] and references therein for other adaptive regression methods.) We also note that the converge rates may vary depending on the choice of basis.

Under Assumptions 1–2, we bound the bias and variance of $\widehat{f}_I(z|\mathbf{x})$ separately.

Lemma 1 (Bias Bound)**.** *Under Assumption 1,*

$$\sum_{i>I} \int (\beta_i(\mathbf{x}))^2 \, d\mathbf{x} = O(I^{-2\beta})$$

Lemma 2 (Variance Bound)**.** *From Assumption 2, it follows that*

$$\sum_{i=1}^{I} \mathbb{E}\left[\int \left(\widehat{\beta}_i(\mathbf{x}) - \beta_i(\mathbf{x})\right)^2 d\mathbf{x}\right] = IO\left(n^{-2\alpha/(2\alpha+d)}\right)$$

Our main result follows.

Theorem 4.1. *Under Assumptions 1 and 2, an upper bound on the risk of the CDE from Equation 2.3 is*

$$\mathbb{E}\left[\iint \left(\widehat{f}_I(z|\mathbf{x}) - f(z|\mathbf{x})\right)^2 dz d\mathbf{x}\right] \leq IO\left(n^{-2\alpha/(2\alpha+d)}\right) + O(I^{-2\beta})$$

See Appendix C for proofs.

Corollary 1. *Under Assumptions 1 and 2, it is optimal to take*

$$I \asymp n^{\frac{2\alpha}{(2\alpha+d)(2\beta+1)}},$$

which yields the rate

$$O\left(n^{-\frac{2\beta}{2\beta+d\frac{2\beta+1}{2\alpha}+1}}\right)$$

for the estimator in Equation 2.3.

To summarize: The convergence rate of FlexCode only depends on d, the "true" dimension of the problem. Moreover, the rate is near minimax with regards to d: In the isotropic setting where $\mathbf{x}$ and z have the same degree of smoothness, i.e., $\alpha = \beta$, the rate becomes

$$O\left(n^{-\frac{2\alpha}{2\alpha+d\frac{2\alpha+1}{2\alpha}+1}}\right),$$

which is close to the minimax rate $O\left(n^{-\frac{2\alpha}{(2\alpha+1+d)}}\right)$ of a conditional density estimator with d covariates [28]. The difference is the multiplicative factor $\frac{2\alpha+1}{2\alpha}$, which gets closer to 1, the smoother f is. Although FlexCode's rate is slightly slower than the optimal rate,[6] the estimator is still considerably faster than $O\left(n^{-\frac{2\alpha}{(2\alpha+1+D)}}\right)$, the usual minimax rate of a nonparametric conditional density estimator in $\mathbb{R}^D$. In other words, even though there are D covariates, our estimator can overcome the curse-of-dimensionality and behave as if there are only $d \ll D$ covariates.

Finally, note that although we here restrict our examples to cases where either (i) the intrinsic dimension is small or (ii) several covariates are irrelevant, the theory we develop can easily be applied to other settings for high-dimensional regression estimation. For instance, Yang and Tokdar [57] introduce a third type of sparse structure: in their paper, r may depend on all D covariates, but admits an additive structure $r = \sum_{s=1}^{k} r_s$, where each component function r_s depends on a small number d_s of predictors. The authors then show that an additive Gaussian process regression achieves good rates of convergence in such a setting. It follows that FlexCode can achieve good rates under a similar additive setting $f(z|\mathbf{x}) = \sum_{s=1}^{k} f_s(z|\mathbf{x})$ if one estimates the expansion coefficients via additive Gaussian process regression.

5. Conclusions

With FlexCode, one can use *any* regression methodology to estimate a conditional density. In other words, FlexCode is a powerful inference and data analysis tool that converts prediction to the problem of understanding the role of covariates in *explaining* the outcome, with meaningful measures of uncertainty attached to the predictions. Because of the flexibility of the method, one can construct estimators for a range of different scenarios with complex, high-dimensional data. In the paper, we emphasized examples where several redundant covariates are correlated, and examples where only a small number of covariates influence the distribution of the response. We showed that FlexCode has good theoretical properties and empirical performance comparable to state-of-the-art approaches in a wide variety of settings, including cases with mixed data types and functional data.

[6] The reason may be that that we optimize the tuning parameters of each regression $\widehat{\beta}_i(\mathbf{x})$ so as to have optimal regression estimates (Assumption 2) rather than an optimal estimate of $f(z|\mathbf{x})$.

In the paper, we restricted most analyses to Fourier bases in the outcome space, but for distributions that are inhomogeneous with respect to the response variable, one may benefit from nonlinear approximations in a wavelet basis [37]. We will explore this aspect further in a separate paper, as well as extensions of FlexCode to approximate likelihood computation for structured data and complex simulation models. Another interesting direction for future work is variable selection via FlexCode. For example, FlexCode-Forest and FlexCode-SAM currently perform a separate variable selection for each coefficient $\beta_i(\mathbf{x})$ in FlexCode (Eq. 2.2), but one can unify these results to define a common support for the final FlexCode estimate.

Appendix A: Diagnostic test of conditional density estimates

To assess how well a model actually fits the observed data, we use coverage plots that are based on *Highest-Predictive Density (HPD) regions*.

Let $\widehat{f}_{z|\mathbf{x}_k}$ denote the estimated conditional density function for z given $\mathbf{x}_k$. For every α in a grid of values in $\{\alpha_1, \ldots, \alpha_B\} \subseteq [0,1]$ and for every data point k in the test sample, we define a set $A_{i,k}$ such that $\int_{A_{i,k}} \widehat{f}(z|\mathbf{x}_k)dz = \alpha_i$. Here we choose the set $A_{i,k}$ with the smallest area: $A_{i,k} = \{z : f(z|\mathbf{x}_k) > t\}$ where t is such that $\int_{A_{i,k}} \widehat{f}(z|\mathbf{x}_k)dz = \alpha_i$; i.e., $A_{i,k}$ is a Highest Predictive Density region.

For each $i = 1, \ldots, B$, let $\widehat{\alpha}_i = \frac{1}{n}\sum_{k=1}^{n} \mathbb{I}(Z_k \in A_{i,k})$. If $\widehat{f}_{z|\mathbf{x}}$ and the true density $f_{z|\mathbf{x}}$ are similar, then $\widehat{\alpha}_i \approx \alpha_i$. Hence, as a diagnostic tool, we graph $\widehat{\alpha}_i$ versus α_i for the test set, and assess how close these points are to the line $\widehat{\alpha} = \alpha$. For each α_i, we also include a 95% confidence interval based on a normal approximation to the binomial distribution.

Appendix B: Additional Twitter data

Here we consider 5000 geotagged tweets posted in July 2015 that include either the keyword *frio* or the keyword *calor*; these words mean cold and hot in Spanish as well as in Portuguese. As in Sec. 3.3, the goal is to predict the latitude and longitude of a tweet, $\mathbf{z}$, based on its content $\mathbf{x}$. Using the same methodology (*FlexCode-SAM*) as before, we estimate $f(\mathbf{z}|\mathbf{x})$.

Fig. 9 shows the results for three tweets. In the tweet corresponding to the left plot, the user mentions "beach" and "heat". Because (i) July is a summer month in the north hemisphere, (ii) the tweet is in Spanish, and (iii) it mentions "beach", FlexCode automatically assigns high probability to the coast of Spain. For the example corresponding to the middle plot, on the other hand, the word "beach" does not occur, but the tweet is in Spanish and it mentions hot weather. As a result, our density model assigns high probability to the interior of Spain. Our final example, corresponding to the right plot in the figure, is a tweet in Portuguese about cold weather. Our FlexCode model here assigns high probability to big cities in Brazil, which is consistent with July being a winter month in the south hemisphere. We also notice that it in the winter rains a lot

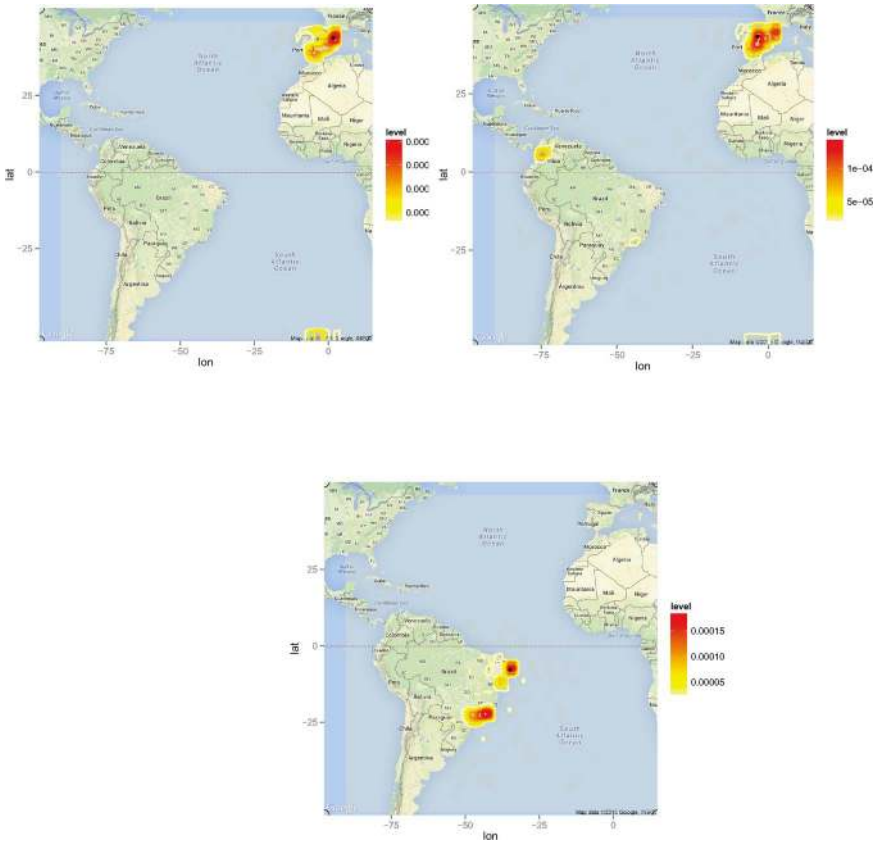


FIG 9. *Level sets of the estimated probability densities of the location of three tweets given their contents. The black dots show the true location where each tweet was posted. Left:* Contra la ola de calor... Un chapuzón en la playa tempranito y ahora... Reclusión en casa... *(Against the heat wave... A dip in the beach very early and now... Confinement at home). Middle:* Combatiendo el calor #verano #lacuevadekrusty # elmolar *(Fighting the heat #summer #lacuevadekrusty #elmolar). Right:* Domingo de chuva, frio gostoso é dia de: Fazer planilha do DVD kkkkk *(Rainy Sunday, pleasant cold is a day of: Making a DVD playlist lol).*

in Recife, the northernmost city that are colored red in the density plot. This is why FlexCode assigns a high probability to this location despite the city being much smaller than Sao Paulo and Rio de Janeiro.

Appendix C: Proofs and additional results

To prove that the estimators in examples E1–E6 in Sec. 4 satisfy Assumption 2, we only need to show that smoothness in the conditional density $f(z|\mathbf{x})$ (seen as a function of $\mathbf{x}$) implies smoothness for each varying coefficient $\beta_i(\mathbf{x})$. Assumption 2 then follows directly from known convergence results for regression. For E3 and E4, note that if there exists a subset $R \subseteq \{1, \ldots, D\}$ with $|R| = d$

such that $f(z|\mathbf{x}) = f(z|(x_i)_{i\in R})$ (i.e., there are only d relevant covariates), then $\beta_i(\mathbf{x}) = \beta_i((x_i)_{i\in R})$.

Different estimators use different notions of smoothness. In Kpotufe [33], the authors show that k-NN regressors converge at rates the depend only on the intrinsic dimension of data if the target function is Lipschitz. Hence, for example E1, we use the Lipschitz notion of smoothness:

Lemma 3. *Let $\{\phi_i\}_i$ be the Fourier basis. If, for every fixed $z \in \mathbb{R}$, $f(z|\mathbf{x})$ is L-Lipschitz function, then $\beta_i(\mathbf{x})$ is $\sqrt{2}L$-Lipschitz for all $i \in \mathbb{N}$.*

Proof. Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^D$. Then

$$\begin{aligned}|\beta_i(\mathbf{x}) - \beta_i(\mathbf{y})| &= \left|\int \phi_i(z) f(z|\mathbf{x}) dz - \int \phi_i(z) f(z|\mathbf{y}) dz\right| \\ &\le \int |\phi_i(z)|\, |f(z|\mathbf{x}) - f(z|\mathbf{y})|\, dz \\ &\le L||\mathbf{x}-\mathbf{y}|| \int |\phi_i(z)|\, dz \le \sqrt{2}L||\mathbf{x}-\mathbf{y}|| \int |\phi_i(z)|^2\, dz \\ &= \sqrt{2}L||\mathbf{x}-\mathbf{y}||\end{aligned}$$

□

Local polynomial regression [4] and Rodeo [35] use the notion of bounded partial derivatives. Hence, we use the following result:

Lemma 4. *Let $\{\phi_i\}_i$ be the Fourier basis. If for every fixed $z \in \mathbb{R}$, $f(z|\mathbf{x})$ has all partial derivatives of order p bounded by K, then $\beta_i(\mathbf{x})$ has all partial derivatives of order p bounded by $\sqrt{2}K$*

Proof. Let $\mathbf{x} \in \mathbb{R}^D$ and $a_1, \ldots, a_p \in \{1, 2, \ldots, D\}$. Then

$$\begin{aligned}\left|\frac{\partial}{\partial x_{a_1} \ldots \partial x_{a_p}} \beta_i(\mathbf{x})\right| &= \left|\frac{\partial}{\partial x_{a_1} \ldots \partial x_{a_p}} \int \phi_i(z) f(z|\mathbf{x}) dz\right| \\ &\le \int |\phi_i(z)| \left|\frac{\partial}{\partial x_{a_1} \ldots \partial x_{a_p}} f(z|\mathbf{x})\right| dz \\ &\le \sqrt{2}K\end{aligned}$$

□

The notion of smoothness in Bertin and Lecué [2] is based on Hölderian classes. Hence:

Lemma 5. *Let $\{\phi_i\}_i$ be the Fourier basis and $\mathcal{P}_l(f)(\cdot, \mathbf{x})$ be Taylor polynomial of order l associated with f at the point $\mathbf{x}$. If, for every fixed $z \in \mathbb{R}$, $f_z(\mathbf{x}) := f(z|\mathbf{x})$ belongs to $\Sigma(\alpha, L)$, the α-Hölderian class, i.e., $|f_z(\mathbf{x}) - \mathcal{P}_l(f_z)(\mathbf{t}, \mathbf{x})| \le L||\mathbf{t}-\mathbf{x}||_1^\alpha$ where $l = \lfloor \alpha \rfloor$, then $\beta_i(\mathbf{x})$ belongs to $\Sigma(\alpha, \sqrt{2}L)$ for all $i \in \mathbb{N}$.*

Proof. Because $\beta_i(\mathbf{x}) = \int \phi_i(z) f(z|\mathbf{x}) dz$, then $\mathcal{P}_l(\beta_i)(\mathbf{t}, \mathbf{x}) = \int \phi_i(z) \mathcal{P}_l(f_z)(\mathbf{t}, \mathbf{x}) dz$. Hence, we have that

$$|\beta_i(\mathbf{x}) - \mathcal{P}_l(\beta_i)(\mathbf{t}, \mathbf{x})| \le \int |\phi_i(z)|\, |f(z|\mathbf{x}) - \mathcal{P}_l(f_z)(\mathbf{t}, \mathbf{x})| dz \le \sqrt{2}L||\mathbf{t}-\mathbf{x}||_1^\alpha$$

□

The spectral series estimator [36] assumes that the regression function is smooth with respect to P. Hence:

Lemma 6. *Let $\{\phi_i\}_i$ be the Fourier basis and assume that, for every fixed $z \in \mathbb{R}$, $\int ||\nabla f(z|\mathbf{x})||^2 dS(\mathbf{x}) < \infty$. Then, for all $i \in \mathbb{N}$, $\int ||\nabla \beta_i(\mathbf{x})||^2 dS(\mathbf{x}) < \infty$.*

Proof. Because $\beta_i(\mathbf{x}) = \int \phi_i(z) f(z|\mathbf{x}) dz$, then

$$\begin{aligned}
\int ||\nabla \beta_i(\mathbf{x})||^2 dS(\mathbf{x}) &= \int \left\| \nabla \int \phi_i(z) f(z|\mathbf{x}) dz \right\|^2 dS(\mathbf{x}) \\
&= \int \left\| \int \phi_i(z) \nabla f(z|\mathbf{x}) dz \right\|^2 dS(\mathbf{x}) \\
&\leq \int \left(\int \phi_i^2(z) dz \right) \left(\int ||\nabla f(z|\mathbf{x})||^2 dz \right) dS(\mathbf{x}) \\
&= \int \left(\int ||\nabla f(z|\mathbf{x})||^2 dS(\mathbf{x}) \right) dz < \infty
\end{aligned}$$

□

Finally, the local linear functional regression estimator [1] assumes that the regression function has continuous second derivatives. Hence:

Lemma 7. *Let $\{\phi_i\}_i$ be the Fourier basis and assume that $\mathbf{x} \in \mathcal{L}^2([0,1])$ and that, for every fixed $z \in \mathbb{R}$, $f(z|\mathbf{x})$ has continuous second derivative. Then $\beta_i(\mathbf{x})$ also has continuous second derivative for every $i \in \mathbb{N}$.*

Proof. Because $\beta_i(\mathbf{x}) = \int \phi_i(z) f(z|\mathbf{x}) dz$, then

$$\frac{d^2 \beta_i(\mathbf{x})}{d\mathbf{x}^2} = \int \phi_i(z) \frac{d^2 f(z|\mathbf{x})}{d\mathbf{x}^2} dz$$

□

We now present the proofs of the other results presented in the paper.

C.1. Proof of Lemma 1

Proof. Because $f(z|\mathbf{x})$ belongs to $W_\phi(s_\mathbf{x}, c_\mathbf{x})$ for all z, and $f(z|\mathbf{x}) = \sum_{i \geq 1} \beta_i(\mathbf{x}) \phi_i(z)$, we have that

$$\sum_{i \geq I} I^{2s_\mathbf{x}} \left(\beta_i(\mathbf{x})\right)^2 \leq \sum_{i \geq I} i^{2s_\mathbf{x}} \left(\beta_i(\mathbf{x})\right)^2 \leq c_\mathbf{x}^2.$$

Hence

$$\sum_{i \geq I} \int \left(\beta_i(\mathbf{x})\right)^2 d\mathbf{x} \leq \int \frac{c_\mathbf{x}^2}{I^{2s_\mathbf{x}}} d\mathbf{x} = O(I^{-2\beta}).$$

□

C.2. Proof of Theorem 1

Proof.

$$\iint \left(\widehat{f}_I(z|\mathbf{x}) - f(z|\mathbf{x}) \right)^2 dz d\mathbf{x} =$$

$$\iint \left(\sum_{i=1}^{I} \widehat{\beta}_i(\mathbf{x})\phi_i(z) - \sum_{i\geq 1} \beta_i(\mathbf{x})\phi_i(z) \right)^2 dz d\mathbf{x} =$$
$$\iint \left(\sum_{i=1}^{I} (\widehat{\beta}_i(\mathbf{x}) - \beta_i(\mathbf{x}))\phi_i(z) - \sum_{i>I} \beta_i(\mathbf{x})\phi_i(z) \right)^2 dz d\mathbf{x} \overset{(*)}{=}$$
$$\int \left(\sum_{i=1}^{I} (\widehat{\beta}_i(\mathbf{x}) - \beta_i(\mathbf{x}))^2 + \sum_{i>I} (\beta_i(\mathbf{x}))^2 \right) d\mathbf{x} =$$
$$\sum_{i=1}^{I} \int (\widehat{\beta}_i(\mathbf{x}) - \beta_i(\mathbf{x}))^2 d\mathbf{x} + \sum_{i>I} \int (\beta_i(\mathbf{x}))^2 d\mathbf{x},$$

where step $(*)$ follows from expanding the square and the fact that the Fourier basis is orthonormal (i.e., the cross products in the expansion are zero).

The final result follows from Lemmas 1 and 2. □

References

[1] A. Baíllo and A. Grané. Local linear regression for functional predictor and scalar response. *Journal of Multivariate Analysis*, 100(1):102–111, 2009. MR2460480

[2] K. Bertin and G. Lecué. Selection of variables and dimension reduction in high-dimensional non-parametric regression. *Electronic Journal of Statistics*, 2:1224–1241, 2008. MR2461900

[3] K. Bertin, C. Lacour, and V. Rivoirard. Adaptive pointwise estimation of conditional density function. In *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, volume 52, pages 939–980. Institut Henri Poincaré, 2016. MR3498017

[4] P. J. Bickel and B. Li. Local polynomial regression on unknown manifolds. In *IMS Lecture Notes–Monograph Series, Complex Datasets and Inverse Problems*, volume 54, pages 177–186. Institute of Mathematical Statisitcs, 2007. MR2459188

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. F Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.

[7] A. Desai, H. Singh, and V. Pudi. Gear: Generic, efficient, accurate kNN-based regression. In *Proc. Int. Conf. KDIR*, pages 1–13, 2010.

[8] M. Di Marzio, S. Fensore, A. Panzera, and C. C. Taylor. A note on nonparametric estimation of circular conditional densities. *Journal of Statistical Computation and Simulation*, pages 1–10, 2016. MR3511014

[9] S. Efromovich. *Nonparametric Curve Estimation: Methods, Theory and Application*. Springer, 1999. MR1705298

[10] S. Efromovich. Dimension reduction and adaptation in conditional density estimation. *Journal of the American Statistical Association*, 105(490):761–774, 2010. MR2724859

[11] A. E Evrard, J. Bialek, M. Busha, M. White, S. Habib, et al. Virial scaling of massive dark matter halos: why clusters prefer a high normalization cosmology. *The Astrophysical Journal*, 672(1):122, 2008.

[12] J. Fan, Q. Yao, and H. Tong. Estimation of conditional densities and sensitivity measures in nonlinear dynamical systems. *Biometrika*, 83(1):189–206, 1996. MR1399164

[13] J. Fan, L. Peng, Q. Yao, and W. Zhang. Approximating conditional density functions using dimension reduction. *Acta Mathematicae Applicatae Sinica*, 25(3):445–456, 2009. MR2506985

[14] Y. Fan, D. J. Nott, and S. A. Sisson. Approximate bayesian computation via regression density estimation. *Stat*, 2(1):34–48, 2013.

[15] A. Fernández-Soto, K. M. Lanzetta, H. W. Chen, B. Levine, and N. Yahata. Error analysis of the photometric redshift technique. *Monthly Notices of the Royal Astronomical Society*, 330:889–894, 2001.

[16] F. Ferraty and P. Vieu. *Nonparametric functional data analysis: theory and practice*. Springer Science & Business Media, 2006. MR2229687

[17] F. Ferraty, A. Mas, and P. Vieu. Nonparametric regression on functional data: inference and practical aspects. *Australian & New Zealand Journal of Statistics*, 49(3):267–286, 2007. MR2396496

[18] P. E. Freeman, R. Izbicki, and A. B. Lee. A unified framework for constructing, tuning and assessing photometric redshift density estimates in a selection bias setting. *Monthly Notices of the Royal Astronomical Society*, 468(4):4556–4565, 2017.

[19] A. G. Gray and A. W. Moore. Nonparametric density estimation: Toward computational tractability. In *SIAM Data Mining*, pages 203–211, 2003.

[20] P. Hall, J. S. Racine, and Q. Li. Cross-validation and the estimation of conditional probability densities. *Journal of the American Statistical Association*, 99:1015–1026, 2004. MR2109491

[21] T. Hastie and R. Tibshirani. Varying-coefficient models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 757–796, 1993. MR1229881

[22] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. New York: Springer-Verlag, 2001. MR1851606

[23] T. Hayfield and J. S. Racine. Nonparametric econometrics: The np package. *Journal of Statistical Software*, 27(5), 2008.

[24] N. J. Higham. Computing the nearest correlation matrix – a problem from finance. *IMA journal of Numerical Analysis*, 22(3):329–343, 2002. MR1918653

[25] M. P. Holmes, A. G. Gray, and C. L. Jr. Isbell. Fast nonparametric conditional density estimation, 2007.

[26] R. J. Hyndman, D. M. Bashtannyk, and G. K. Grunwald. Estimating and

visualizing conditional densities. *Journal of Computational & Graphical Statistics*, 5:315–336, 1996. MR1422114

[27] T. Ichimura and D. Fukuda. A fast algorithm for computing least-squares cross-validations for nonparametric conditional kernel density functions. *Computational Statistics Data Analysis*, 54(12):3404–3410, 2010. MR2727763

[28] R. Izbicki and A. B. Lee. Nonparametric conditional density estimation in a high-dimensional regression setting. *Journal of Computational and Graphical Statistics*, 25(4):1297–1316, 2016. MR3572041

[29] R. Izbicki, A. B. Lee, and C. M. Schafer. High-dimensional density ratio estimation with extensions to approximate likelihood computation. *Journal of Machine Learning Research (AISTATS Track)*, pages 420–429, 2014.

[30] R. Izbicki, A. B. Lee, and P. E. Freeman. Photo-z estimation: An example of nonparametric density estimation under selection bias for multivariate data. *The Annals of Applied Statistics, to appear*, 2016.

[31] A. Kalda and S. Siddiqui. Nonparametric conditional density estimation of short-term interest rate movements: procedures, results and risk management implications. *Applied Financial Economics*, 23(8):671–684, 2013.

[32] M. C. Kind and R. J. Brunner. Tpz: photometric redshift pdfs and ancillary information by using prediction trees and random forests. *Monthly Notices of the Royal Astronomical Society*, 432(2):1483–1501, 2013.

[33] S. Kpotufe. k-nn regression adapts to local intrinsic dimension. In *Advances in Neural Information Processing Systems*, pages 729–737, 2011.

[34] S. Kpotufe and S. Dasgupta. A tree-based regressor that adapts to intrinsic dimension. *Journal of Computer and System Sciences*, 78(5):1496–1515, 2012. MR2926146

[35] J. Lafferty and L. Wasserman. Rodeo: sparse, greedy nonparametric regression. *The Annals of Statistics*, 36(1):28–63, 2008. MR2387963

[36] A. B. Lee and R. Izbicki. A spectral series approach to high-dimensional nonparametric regression. *Electronic Journal of Statistics*, 10(1):423–463, 2016. MR3466189

[37] S. Mallat. *A wavelet tour of signal processing*. Academic press, 1999. MR1614527

[38] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[39] L. Meier, S. Van de Geer, and P. Bühlmann. High-dimensional additive modeling. *The Annals of Statistics*, 37(6B):3779–3821, 2009. MR2572443

[40] M. Ntampaka, H. Trac, D. J. Sutherland, N. Battaglia, B. Poczos, and J. Schneider. A machine learning approach for dynamical mass measurements of galaxy clusters. *The Astrophysical Journal*, 803(2):50, 2015.

[41] M. Ntampaka, H. Trac, D. J. Sutherland, S. Fromenteau, B. Póczos, and J. Schneider. Dynamical mass measurements of contaminated galaxy clusters using machine learning. *arXiv preprint arXiv:1509.05409*, 2015.

[42] G. Papamakarios and I. Murray. Fast ϵ-free inference of simulation models with bayesian conditional density estimation. *arXiv preprint arXiv:1605.06376*, 2016.

[43] A. Quintela-del Río, F. Ferraty, and P. Vieu. Nonparametric conditional density estimation for functional data. econometric applications. In *Recent Advances in Functional Data Analysis and Related Topics*, pages 263–268. Springer, 2011. MR2815592

[44] M. M. Rau, S. Seitz, F. Brimioulle, E. Frank, O. Friedrich, D. Gruen, and B. Hoyle. Accurate photometric redshift probability density estimation — method comparison and application. *Monthly Notices of the Royal Astronomical Society*, 452(4):3710–3725, 2015.

[45] P. Ravikumar, J. Lafferty, H. Liu, and L. Wasserman. Sparse additive models. *Journal of the Royal Statistical Society, Series B*, 71(5):1009–1030, 2009. MR2750255

[46] V. C. Raykar. Scalable machine learning for massive datasets: Fast summation algorithms. 2007.

[47] E. Rodrigues, R. Assunção, G. L. Pappa, D. Renno, and W. Meira Jr. Exploring multiple evidence to infer users' location in twitter. *Neurocomputing*, 2015. URL http://www.sciencedirect.com/science/article/pii/S092523121500764X.

[48] M. Rosenblatt. Conditional probability density and regression estimators. In P. R. Krishnaiah, editor, *Multivariate Analysis II*. 1969. MR0254987

[49] E. S. Sheldon, C. E. Cunha, R. Mandelbaum, J. Brinkmann, and B. A. Weaver. Photometric redshift probability distributions for galaxies in the SDSS DR8. *The Astrophysical Journal Supplement Series*, 201(2), 2012.

[50] M. Shiga, V. Tangkaratt, and M. Sugiyama. Direct conditional probability density estimation with sparse feature selection. *Machine Learning*, 100(2–3):161–182, 2015. MR3383968

[51] M. Sugiyama, I. Takeuchi, T. Suzuki, T. Kanamori, H. Hachiya, and D. Okanohara. Conditional density estimation via least-squares density ratio estimation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 781–788, 2010. MR2895762

[52] D. J. Sutherland, L. Xiong, B. Póczos, and J. Schneider. Kernels on sample sets via nonparametric divergence estimates. *arXiv preprint arXiv:1202.0302*, 2012.

[53] I. Takeuchi, K. Nomura, and T. Kanamori. Nonparametric conditional density estimation using piecewise-linear solution path of kernel quantile regression. *Neural Computation*, 21(2):533–559, 2009. MR2477869

[54] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996. MR1379242

[55] Q. Wang, S. R. Kulkarni, and S. Verdú. A nearest-neighbor approach to estimating divergence between continuous random vectors. In *2006 IEEE International Symposium on Information Theory*, pages 242–246, 2006.

[56] L. Wasserman. *All of Nonparametric Statistics*. Springer-Verlag New York, Inc., 2006. MR2172729

[57] Y. Yang and S. T. Tokdar. Minimax-optimal nonparametric regression in high dimensions. *The Annals of Statistics*, 43(2):652–674, 2015. MR3319139

[58] Y. Zhang, J. Duchi, and M. Wainwright. Divide and conquer kernel ridge regression. In *Conference on Learning Theory*, pages 592–617, 2013. MR3450540

[59] L. Zhao and Z. Liu. Strong consistency of the kernel estimators of conditional density function. *Acta Mathematica Sinica*, 1(4):314–318, 1985. MR0867902