Astronomy and Computing 30 (2020) 100362

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

Conditional density estimation tools in python and R with applications to photometric redshifts and likelihood-free cosmological inference

N. Dalmasso [a,*], T. Pospisil [a,1], A.B. Lee [a], R. Izbicki [b], P.E. Freeman [a], A.I. Malz [c]

[a] Department of Statistics & Data Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
[b] Department of Statistics, Federal University of São Carlos, São Paulo, Brazil
[c] Center for Cosmology and Particle Physics, New York University, New York, NY 10003, USA

ARTICLE INFO

Article history:
Received 1 September 2019
Accepted 30 December 2019
Available online 13 January 2020

Keywords:
Nonparametric statistics
Statistical software
Statistical computing
Methods: Data analysis
Galaxies: Distances and redshifts
Cosmology: Cosmological parameters

ABSTRACT

It is well known in astronomy that propagating non-Gaussian prediction uncertainty in photometric redshift estimates is key to reducing bias in downstream cosmological analyses. Similarly, likelihood-free inference approaches, which are beginning to emerge as a tool for cosmological analysis, require a characterization of the full uncertainty landscape of the parameters of interest given observed data. However, most machine learning (ML) or training-based methods with open-source software target point prediction or classification, and hence fall short in quantifying uncertainty in complex regression and parameter inference settings such as the applications mentioned above. As an alternative to methods that focus on predicting the response (or parameters) **y** from features **x**, we provide nonparametric conditional density estimation (CDE) tools for approximating and validating the entire probability density function (PDF) $p(\mathbf{y}|\mathbf{x})$ of **y** given (i.e., conditional on) **x**. This density approach offers a more nuanced accounting of uncertainty in situations with, e.g., nonstandard error distributions and multimodal or heteroskedastic response variables that are often present in astronomical data sets. As there is no one-size-fits-all CDE method, and the ultimate choice of model depends on the application and the training sample size, the goal of this work is to provide a comprehensive range of statistical tools and open-source software for nonparametric CDE and method assessment which can accommodate different types of settings – involving, e.g., mixed-type input from multiple sources, functional data, and images – and which in addition can easily be fit to the problem at hand. Specifically, we introduce four CDE software packages in `Python` and `R` based on ML prediction methods *adapted and optimized for CDE*: `NNKCDE`, `RFCDE`, `FlexCode`, and `DeepCDE`. Furthermore, we present the `cdetools` package with evaluation metrics. This package includes functions for computing a CDE loss function for tuning and assessing the quality of individual PDFs, together with diagnostic functions that probe the population-level performance of the PDFs. We provide sample code in `Python` and `R` as well as examples of applications to photometric redshift estimation and likelihood-free cosmological inference via CDE.

1. Introduction

Machine learning (ML) has seen a surge in popularity in almost all fields of astronomy that involve massive amounts of complex data (Ball and Brunner, 2010; Way et al., 2012; Ivezić et al., 2014; Ntampaka et al., 2019). Most ML methods target regression and classification, whose primary goal is to return a point estimate of an unknown response variable **y** given observed features **x**, often falling short in quantifying nontrivial uncertainty in **y**. For instance, returning a point estimate for a supernova's type **y** given a supernova's light curve **x**, or for a galaxy mass **y** given its light spectrum **x**, fails to capture degeneracies in the mapping from **x** to **y**. Neglecting to propagate these uncertainties through down-stream analyses may lead to imprecise or even inaccurate inferences of physical parameters.

Consider the following two examples of problems where uncertainty quantification can be impactful:

* Corresponding author.
E-mail addresses: ndalmass@stat.cmu.edu (N. Dalmasso), popt23@gmail.com (T. Pospisil), annlee@cmu.edu (A.B. Lee), rafaelizbicki@gmail.com (R. Izbicki), pfreeman@cmu.edu (P.E. Freeman), aimalz@nyu.edu (A.I. Malz).
[1] Present Address: Google LLC, 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA

https://doi.org/10.1016/j.ascom.2019.100362
2213-1337/

- **Photometric redshift estimation.** In photometric redshift (photo-z) estimation, one attempts to constrain the cosmological *redshift* (z) of a galaxy after observing the shifted spectrum using a handful of broadband filters, and sometimes additional variables such as morphology and environment. In a prediction setting, the response $\mathbf{y}$ could be the galaxy's true (i.e. spectroscopically observed) redshift but could also include other galaxy properties (α) such as the galaxy's mass or age; the features $\mathbf{x}$ would represent the collection of directly observable inputs such as photometric magnitudes and colors used to predict $\mathbf{y} = z$ or more generally a multivariate response $\mathbf{y} = (z, \alpha)$. The use of photo-z posterior estimates – that is, estimates of the probability density functions (PDFs) for individual galaxies – is crucial for cosmological inference from photometric galaxy surveys, as $\gtrsim 99\%$ of currently cataloged galaxies are observed solely via photometry, a percentage that will only grow in the coming decade as the Large Synoptic Survey Telescope (LSST) begins gathering data (Ivezić et al., 2019). In addition, widely different redshifts can be consistent with the observed colors of a galaxy; photo-z posterior estimates can capture such degeneracy or multimodality in the distribution whereas point estimates cannot. Though the benefits of using posteriors over α have yet to be fully exploited (Vironen et al., 2018), it is thoroughly established that one can improve down-stream cosmological analysis by properly propagating photo-z estimate uncertainties via probability density functions (PDFs) rather than just using simple point predictions of $\mathbf{y}$ (Mandelbaum et al., 2008; Wittman, 2009; Sheldon et al., 2012; Carrasco Kind and Brunner, 2013; Graff et al., 2014; Schmidt et al., 2020).
- **Forward-modeled observables in cosmology.** Some cosmological probes, such as the type Ia supernova (SN Ia) distance-redshift relationship, have observables that are straightforward to simulate in spite of an intractable likelihood. Likelihood-Free Inference (LFI) methods allow for parameter inference in settings where the likelihood function, which relates the observable data $\mathbf{x}_{\text{obs}}$ to the parameters of interest $\boldsymbol{\theta}$, is too complex to work with directly, but one is able to *simulate* $\mathbf{x}$ from a stochastic forward model at fixed parameter settings $\boldsymbol{\theta}$. The most common approach to LFI or simulation-based parameter inference is Approximate Bayesian Computation (ABC), whose many variations (see Beaumont et al. 2002 and Sisson et al. 2018 for a review) repeatedly draw from the simulator and compare the output with observed data $\mathbf{x}_{\text{obs}}$ in real time to arrive at a set of plausible parameter values consistent with $\mathbf{x}_{\text{obs}}$. With computationally intensive simulations, however, a classical ABC approach may not be practical. An alternative approach to ABC rejection sampling is to use faster training-based methods to assess the uncertainty about $\boldsymbol{\theta}$ for any $\mathbf{x}$ first, and then consider the specific case $\mathbf{x} = \mathbf{x}_{\text{obs}}$.

From a statistical perspective, the right tool for quantifying the uncertainty about $\mathbf{y}$ once $\mathbf{x}$ is observed is the *conditional density* $\text{p}(\mathbf{y}|\mathbf{x})$. In a prediction context such as for photo-z problems, where heteroskedastic errors or multimodal responses may occur, conditional density estimation (CDE) of the density $\text{p}(z|\mathbf{x})$ for the redshift z of individual galaxies given photometric data $\mathbf{x}$ provides a more nuanced accounting of uncertainty than a point estimate or prediction interval alone. CDE can also be used in LFI where, in our notation, the parameters of interest $\boldsymbol{\theta}$ take the role of the "response" y. In such settings, one can apply training-based approaches to forward-simulated data to estimate the posterior probability distribution $\text{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$ of cosmological parameters $\boldsymbol{\theta}$ given observed data $\mathbf{x}_{\text{obs}}$, and from these posteriors then derive, e.g. posterior credible intervals of θ. Section 4.2 shows an example of cosmological parameter inference using mock weak lensing data. Here we follow Izbicki et al. (2014, 2019) to combine ABC and CDE by directly applying CDE techniques (Section 2) and loss functions (Section 3.1) to simulated data $\{(\boldsymbol{\theta}_i, \mathbf{x}_i)\}_{i=1}^n$. Similar works include performing LFI via CDE using Gaussian copulas (Li et al., 2017; Chen and Gutmann, 2019) and random forests (Marin et al., 2016). Other examples include neural density estimation in LFI via mixture density networks and masked autoregressive flows (Papamakarios and Murray, 2016; Lueckmann et al., 2017, 2019; Papamakarios et al., 2019; Greenberg et al., 2019; Alsing et al., 2018, 2019).

Data in astronomy present a challenge to estimating conditional densities, due to both the complexity of the underlying physical processes and the complicated observational noise. Precision cosmology, for example, requires combining data from different scientific probes, each affected by unique sources of systematic uncertainty, to produce samples from complicated joint likelihood functions with nontrivial covariances in a high-dimensional parameter space (Krause et al., 2017; Joudaki et al., 2017; Aghanim et al., 2018; van Uitert et al., 2018; Abbott et al., 2019). In such situations, CDE methods that target a variety of settings and non-standard data (images, correlation functions, mixed data types) become especially valuable. However, for any given data type, there is no one-size-fits-all CDE method. For example, deep neural networks often perform well in settings with large amounts of representative training data but in applications with smaller training samples one may need a different tool. There is also additional value in models that are interpretable and easy to fit to the data at hand.

The goal of this paper is to provide statistical tools and open-source software for nonparametric CDE and method assessment appropriate for challenging data in a variety of inference scenarios.

To our knowledge, existing CDE software either targets discrete $\mathbf{y}$ (e.g., probabilistic classifiers or ordinal classification Frank and Hall, 2001) or uses kernel density estimation (KDE) across all data points to provide an estimate for a continuous $\mathbf{y}$ (Hyndman et al., 2018). What distinguishes our methodology work from others is that we are able to adapt virtually any training-based prediction method to the problem of estimating full probability distributions. By leveraging existing ML methods, we are hence able to provide uncertainty prediction methods and software for more general and complex data settings than 'the computational tools currently available in the literature. In this paper, we showcase and provide `Python` and `R` code for four flexible CDE methods: `NNKCDE`, `RFCDE/fRFCDE`, `FlexCode` and `DeepCDE`.[2] Each CDE approach has particular usage for different settings of response dimensionality, feature types, and computational requirements. Table 1 provides a high-level overview in the top panel, and lists some properties of each method in the bottom panel.

A highlight of our software is that it makes uncertainty quantification straightforward for users of standard open-source machine learning `Python` packages. As `NNKCDE`, `RFCDE/fRFCDE`, `FlexCode` share the `sklearn` (Pedregosa et al., 2011) API (with `fit` and `predict` methods), our methods are usable within the `sklearn` ecosystem for, e.g., cross-validation and model selection. In addition, `FlexCode` is essentially a plug-in method where the user can utilize any `sklearn`-compatible regression function. `DeepCDE` has implementations for both `Tensorflow` (Abadi et al., 2015) and `Pytorch` (Paszke et al., 2019), two of the most widely used deep learning frameworks.

[2] All of these methods, except for `DeepCDE`, have occurred in previously published or archived papers. Hence, we only briefly review the methods in this paper and instead focus on software usage, algorithmic aspects, and the settings under which each method can be applied.

Table 1
Top: Naming convention, high-level summary and hyper-parameters of CDE methods, along with references for further details and code examples. *Bottom:* Comparison of CDE methods in terms of training capacity and compatibility with multivariate response and different types of features, with capacities estimated based on input with around 100 features and a standard i5/i7/quad-core processor with 16GB of RAM. Note that less complex methods (such as `NNKCDE`) tend to be easier to use, easier to interpret, and often perform better in settings with smaller training sets, whereas more complex methods (such as `DeepCDE`) perform better in settings with larger (representative) training sets.

Method	Name	Summary	Hyper-parameters	Details
`NNKCDE`	Nearest Neighbor Kernel CDE	Computes a KDE estimate of multivariate $\mathbf{y}$ using the nearest neighbors of the evaluation point $\mathbf{x}$ in feature space.	• Number of neighbors k • Kernel bandwidth h	Section 2.1 (Code: Appendix A.1)
`RFCDE`	Random Forest CDE	Random forest that partitions the feature space using a CDE loss. Constructs a weighted KDE estimate of multivariate $\mathbf{y}$ with weights defined by leaves in the forest.	• Random forest hyperparams. • Kernel bandwidth h	Section 2.2 (Code: Appendix A.2)
`fRFCDE`	functional Random Forest CDE	RFCDE version suitable for functional features $\mathbf{x}$. Partitions the feature space directly rather than representing $\mathbf{x}$ as a vector.	• Random forest hyperparams. • Kernel bandwidth h • Partition parameter λ	Section 2.2.1 (Code: Appendix A.2.1)
`FlexCode`	Flexible Conditional Density Estimation	Uses basis expansion of univariate y to turn CDE into a series of univariate regression problems.	• Number of expansion coeffs. • Selected regression method hyperparams.	Section 2.3 (Code: Appendix A.3)
`DeepCDE`	Deep Neural Networks CDE	Uses basis expansion of univariate y similar to `FlexCode`, but learns the expansion coefficients simultaneously using a deep neural network.	• Number of expansion coeffs. • Selected deep neural network architecture hyperparams.	Section 2.4 (Code: Appendix A)

Method Complexity ↓

Method	Capacity (# Training Pts)	Multivariate Response	Functional Features	Image Features
`NNKCDE`	Up to $\sim 10^5$	✓		
(f)RFCDE	Up to $\sim 10^6$	✓	✓	
`FlexCode`	Up to $\sim 10^6$		✓	
`DeepCDE`	Up to $\sim 10^8$		✓	✓

In addition to the CDE methods above, we provide the package `cdetools`, which can be used for tuning and assessing the performance of CDE models on held-out validation data. CDE method assessment is challenging per se because we never observe the true conditional probability density, merely samples (observations) from it. Furthermore, whereas loss functions such as the root-mean-squared error (RMSE) are typically used in regression problems, they are not appropriate for the task of uncertainty quantification of estimated probability densities. The `cdetools` package provides two types of functions for method assessment. First, it provides functions for computing a so-called CDE loss function (defined by Eq. (4) in Section 3.1) for tuning and assessing the quality of individual PDFs. Second, it provides diagnostic functions that probe the population-level performance of the PDFs. More specifically, we have included functions for computing the Probability Integral Transform (PIT) and Highest Posterior Density (HPD); these metrics check how well the final density estimates on average fit the data in the tail and highest-density regions, respectively (see Section 3.2, and Fig. 2 for a visual sketch).

Organization of the paper. In Section 2, we introduce tools for conditional density estimation (`NNKCDE`, `RFCDE`/`fRFCDE`, `FlexCode`, `DeepCDE`). In Section 3, we describe tools for model selection and diagnostics. Then, in Section 4, we illustrate our CDE and method assessment tools for three different applications: photo-z estimation, likelihood-free cosmological inference and spec-z estimation. `Python` and `R` code usage examples can be found in Appendix A.

Notation. We denote the true (unknown) conditional density by $p(\mathbf{y}|\mathbf{x})$, whereas an estimate of the density is denoted by $\widehat{p}(\mathbf{y}|\mathbf{x})$. We represent the CDE loss function that measures the discrepancy between the conditional density p and its estimate $\widehat{p}$ by $L(p, \widehat{p})$. Typically this loss cannot be computed directly because it depends on unknown quantities; an estimate of the loss is denoted by $\widehat{L}(p, \widehat{p})$. As before, we continue to use bold-faced letters to denote vectors.

2. Overview of conditional density estimation tools

We start by briefly describing the conditional density estimators in Table 1. Unless otherwise stated, we choose the tuning or hyper-parameters by minimizing the CDE empirical loss in Eq. (5) using cross-validation.

2.1. *NNKCDE*

Nearest-Neighbors Kernel CDE (NNKCDE; Izbicki et al. 2017, Freeman et al. 2017) is a simple and easily interpretable CDE method. It computes a kernel density estimate of $\mathbf{y}$ using the k nearest neighbors of the evaluation point $\mathbf{x}$. The model has only two tuning parameters: the number of nearest neighbors k and the bandwidth h of the smoothing kernel in $\mathbf{y}$-space. Both tuning parameters are chosen in a principled way by minimizing the CDE loss on validation data.

More specifically, the kernel density estimate of $\mathbf{y}$ given $\mathbf{x}$ is defined as

$$\widehat{p}(\mathbf{y}|\mathbf{x}) = \frac{1}{k}\sum_{i=1}^{k} K_h\left[\rho(\mathbf{y}, \mathbf{y}_{s_i(\mathbf{x})})\right], \tag{1}$$

where K_h is a normalized kernel (e.g., a Gaussian function) with bandwidth h, ρ is a distance metric, and $s_i(\mathbf{x})$ is the index of the ith nearest neighbor of $\mathbf{x}$. It is essentially a smoother version of the histogram estimator proposed by Cunha et al. (2009) in that it approximates the density with a smooth continuous function rather than by binning.

We provide the NNKCDE[3] software in both Python and R (Pospisil and Dalmasso, 2019a), with examples in Section Appendix A.1. NNKCDE can accommodate any number of training examples. However, selecting tuning parameters scales quadratically in k, the number of nearest neighbors, which prohibits using $k \gtrsim 10^3$. Our implementation (Izbicki et al. 2019, Appendix D) is computationally more efficient than standard nearest-neighbor kernel smoothers. For instance, we are able to efficiently evaluate the loss function in Eq. (5) on large validation samples by expressing the first integral in terms of convolutions of the kernel function; these have a fast-to-compute closed solution for a Gaussian kernel.

2.2. RFCDE

Random forests (RFs, Breiman 2001) is one of the best off-the-shelf solutions for regression and classification problems. It builds a large collection of decorrelated trees, where each tree is a data-based partition of the feature space. The trees are then averaged to yield a prediction. RFCDE, introduced by Pospisil and Lee (2018), is an extension of random forests to conditional density estimation and multivariate responses. Like NNKCDE, it computes a kernel density estimate of $\mathbf{y}$ but with nearest neighbor weightings defined by the location of the evaluation point $\mathbf{x}$ relative to the leaves in the random forest. RFCDE inherits the advantages of random forests in that it can handle mixed-typed data. It also does not require the user to specify distances or similarities between data points, and it has good performance while remaining relatively interpretable.

The main departure from other random forest algorithms is our criterion for feature space partitioning decisions. In regression contexts, the splitting variable and split point are typically chosen so as to minimize the mean-squared-error loss. In classification contexts, the splits are typically chosen so as to minimize a classification error. Existing random forest density estimation methods such as quantile regression forests by Meinshausen (2006) and the TPZ algorithm by Carrasco Kind and Brunner (2013) use the same tree structure as regression and classification random forests, respectively. RFCDE, however, builds trees that minimize the CDE loss (see Eq. (5)), allowing the forest to adapt to structures in the conditional density; hence overcoming some of the limitations of the usual regression approach for data with heteroskedasticity and multimodality. In addition, RFCDE does not require discretizing the response as in TPZ, thereby providing more accurate results at a lower cost for continuous responses, especially in the case of multivariate continuous responses where binning is problematic. See Pospisil and Lee (2018) for further examples and comparisons.

Another unique feature of RFCDE is that it can handle multivariate responses with joint densities by applying a weighted kernel smoother to $\mathbf{y}$. This added feature enables analysis of complex conditional distributions that describe relationships between multiple responses and features, or equivalently between multiple parameters and observables in an LFI setting. Like quantile regression forests, the RFCDE algorithm takes advantage of the fact that random forests can be viewed as a form of adaptive nearest-neighbor method with the aggregated tree structures determining a weighting scheme. This weighting scheme can then be used to estimate the conditional density $p(\mathbf{y}|\mathbf{x})$, as well as the conditional mean and quantiles, as in quantile regression forests (but for CDE-optimized trees). As mentioned above, RFCDE computes the latter density by a weighted kernel density estimate (KDE) in $\mathbf{y}$ using training points near the evaluation point $\mathbf{x}$. These distances are effectively defined by how often a training point $\mathbf{x}_i$ belongs to the same leaf node as $\mathbf{x}$ in the forest (see Equation 1 in Pospisil and Lee 2018 for details).

Despite the increased complexity of our CDE trees, RFCDE still scales to large data sets because of an efficient computation of splits via orthogonal series. Moreover, RFCDE extends the density estimates on new $\mathbf{x}$ to the multivariate case through the use of multivariate kernel density estimators (Epanechnikov, 1969). In both the univariate and multivariate cases, bandwidth selection can be handled by either plug-in estimators or by tuning using a density estimation loss.

For ease of use in the statistics and astronomy communities, we provide RFCDE[4] in both Python and R, which call a common C++ implementation of the core training functions that can easily be wrapped in other languages (Pospisil and Dalmasso, 2019b).

Remark. Estimating a CDE loss is an inherently harder task than calculating the mean squared error (MSE) in regression. As a consequence, RFCDE might not provide meaningful tree splits in settings with a large number of noisy features. In such settings, one may benefit from combining a regular random forest tree structure (optimized for regression) with a weighted kernel density estimator (for the density calculation). See Section 4.3 for an application to functional data along with software implementation.[5]

2.2.1. fRFCDE

In addition the RFCDE package includes fRFCDE, a variant of RFCDE (Pospisil and Lee, 2019), that can accommodate functional features $\mathbf{x}$ by partitioning in the continuous domain of such features. The spectral energy distribution (SED) of a galaxy is its energy as a function of continuous wavelength λ of light; hence it can be viewed as functional data. Another example of functional data is the shear correlation function of weak lensing, which measures the mean product of the shear at two points as a function of a continuous distance r between those points. Similarly, any function of continuous time is an example of functional data. Treating functional features (like spectra, correlation functions or images) as unordered multivariate vectors on a grid suffers from a curse of dimensionality. As the resolution of the grid becomes finer the dimensionality of the data increases but little additional information is added, due to high correlation between nearby grid points. fRFCDE adapts to this setting by partitioning the domain of each functional feature (or curve) into intervals, and passing the mean values of the function in each interval as input to RFCDE. Feature selection is then effectively done over regions of the domain rather than over single points. More specifically, the partitioning in fRFCDE is governed by the parameter μ of a Poisson process, with each functional feature entering as a high-dimensional vector $\mathbf{x} = (x_1, \ldots, x_d)$. Starting with the first element of the vector, we group the first Poisson(μ) elements together. We then repeat the procedure sequentially until we have assigned all d elements into a group; this effectively partitions the function domain into disjoint intervals $\{(l_i, h_i)\}$. The function mean values or smoothed brightness measurements $\widetilde{x}_i \equiv \int_{l_i}^{h_i} f(\lambda) d\lambda$ of each interval are finally treated as new inputs to a standard (vectorial) RFCDE tree. The splitting of the smoothed predictors $\widetilde{x}_i$ is done independently for each tree in the forest. Other steps of fRFCDE, such as the computation of variable importance, also proceed as in (vectorial) RFCDE but with the averaged values of a region as inputs. As a result, fRFCDE has the capability of identifying the functional inputs and the regions in the input domain that are important for estimating the response

[3] https://github.com/tpospisi/nnkcde.

[4] https://github.com/tpospisi/RFCDE.

[5] Available at https://github.com/Mr8ND/cdetools_applications/spec_z/.

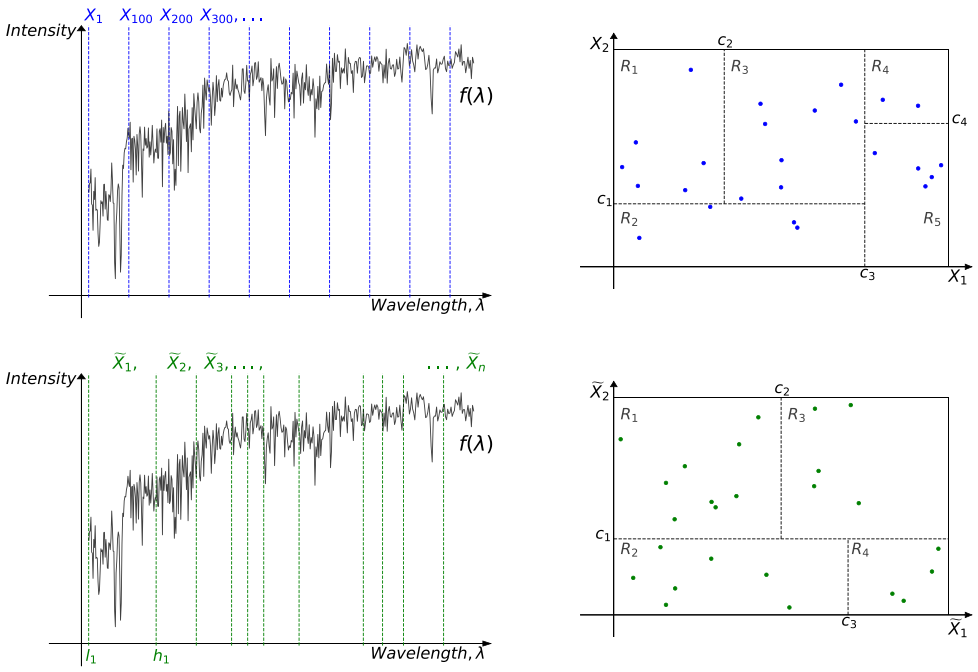

Fig. 1. A schematic diagram of `RFCDE` (top row) and `fRFCDE` (bottom row) applied to a galaxy spectrum from Vanderplas et al. (2012). *Top row*: `RFCDE` treats the intensity x_i at each recorded wavelength λ_i of the spectrum as a feature or "input" to the random forests algorithm — the blue vertical dashed lines indicate every 100th recorded wavelength. `RFCDE` then builds an ensemble of CDE trees, where each tree partitions the feature space according to the CDE loss, as illustrated in the top right figure for features x_1 and x_2. *Bottom row*: `fRFCDE` instead groups nearby measurements together where the group divisions are defined by a Poisson process with parameter μ (vertical green dashed lines, left figure). The new smoothed features $\widetilde{x}_i$ are computed by integrating the intensity over the grouped wavelengths. A forest of CDE trees is then built using the same construction as in `RFCDE` but with the smoothed features as inputs (bottom right figure). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

y. Fig. 1 shows schematically the differences and similarities in construction between standard `RFCDE` and its `fRFCDE` variant.

The `fRFCDE` method is as scalable as standard random forests, accommodating training sets on the order of 10^6. As the examples in Section 4.3 show, we can obtain substantial gains with a functional approach, both in terms of statistical performance (that is, CDE loss) and computational time. In addition, the change in code syntax is minimal, as one only needs to pass the Poisson μ parameter as the `flambda` argument during the forest initialization. Examples in `Python` and `R` are provided in Appendix A.2.1.

2.3. `FlexCode`

Introduced by Izbicki and Lee (2017), `FlexCode`[6] is a CDE method that uses a basis expansion for the univariate response y and poses CDE as a series of univariate regression problems. The main advantage of this method is its flexibility as any regression method can be applied toward CDE, enabling us to tailor our choice of regression method to the intrinsic structure and type of data at hand.

More precisely, let $\{\phi_j(y)\}_j$ be an orthonormal basis like a Fourier or wavelet basis for functions of $y \in \mathbb{R}$. The key idea of `FlexCode` is to express the unknown density $p(y|\mathbf{x})$ as a basis expansion

$$p(y|\mathbf{x}) = \sum_j \beta_j(\mathbf{x})\phi_j(y). \tag{2}$$

By the orthogonality property of the basis, the (unknown) expansion coefficients $\{\beta_j(\mathbf{x})\}_j$ are then just orthogonal projections of $p(y|\mathbf{x})$ onto the basis vectors. We can estimate these coefficients using a training set of $(\mathbf{x}, y)$ data by *regressing* the transformed response variables $\phi_j(y)$ on predictors $\mathbf{x}$ for every basis function j (see Izbicki and Lee, 2017 Equation 2.2, for details). The number of basis function n_{basis} is chosen by minimizing a CDE loss function on validation data. The estimated density, $\sum_{j=1}^{n_{basis}} \widehat{\beta}_j(\mathbf{x})\phi_j(y)$, may contain small spurious bumps induced by the Fourier approximation and may not integrate to one. We remove such artifacts as described in Izbicki and Lee (2016) by applying a thresholding parameter δ chosen via cross-validation. `FlexCode` turns a challenging density estimation problem into a simpler regression problem, where we can choose any regression method that fits the problem at hand.

To provide a concrete example, for high-dimensional $\mathbf{x}$ (such as galaxy spectra) we can use methods such as sparse or spectral series ("manifold") regression methods (Tibshirani, 1996; Ravikumar et al., 2009; Lee and Izbicki, 2016); see Section 4.3 for an example with `FlexCode-Series`. For multi-probe studies with mixed data types, we can use random forests regression (Breiman, 2001). On the other hand, large-scale photometric galaxy surveys such as LSST require methods that can work with data from millions, if not billions, of galaxies. Schmidt et al. (2020) present the results of an initial study of the LSST Dark Energy Science Collaboration (LSST-DESC). Their initial data challenge ("Photo-z DC 1") compares the CDEs of a dozen photo-z codes run on simulations of LSST galaxy photometry catalogs in the presence of complete, correct, and representative training data. `FlexZBoost`, a version of `FlexCode` based on the scalable gradient boosting regression technique by Chen and Guestrin (2016), was entered into the data challenge because of the method's ability to scale to massive data. In the DC1 analysis, `FlexZBoost` was among the strongest performing codes according to established performance metrics of such PDFs and was one of only two codes to beat the experimental control under a more discriminating metric, the CDE loss.

[6] https://github.com/tpospisi/FlexCode (`Python`) and https://github.com/rizbicki/FlexCoDE (`R`).

For massive surveys such as LSST, `FlexCode` also has another advantage compared to other CDE methods, namely its compact, lossless storage format. Juric et al. (2017) establishes that LSST has allocated ∼100 floating point numbers to quantify the redshift of each galaxy. As is shown in Schmidt et al. (2020), the myriad methods for deriving photo-z PDFs yield radically different results, motivating a desire to store the results of more than one algorithm in the absence of an obvious best choice. For the photo-z PDFs of most codes, one may need to seek a clever storage parameterization to meet LSST's constraints (Carrasco Kind and Brunner, 2014; Malz et al., 2018), but `FlexCode` is virtually immune to this restriction. Since `FlexCode` relies on a basis expansion, one only needs to store n_{basis} coefficients per target density for a lossless compression of the estimated PDF with no need for binning. Indeed, for DC1, we can with `FlexZBoost` reconstruct our estimate $\widehat{p}(z|\mathbf{x})$ at any resolution from estimates of the first 35 coefficients in a Fourier basis expansion. In other words, `FlexZBoost` enables the creation and storage of high-resolution photo-z catalogs for several billion galaxies at no added cost.

Our public implementation of `FlexCode`– available in both `Python` (Pospisil et al., 2019) and `R` (Izbicki and Pospisil, 2019), respectively – cross-validates over regression tuning parameters (such as the number of nearest neighbors k in `FlexCode-kNN`) and computes the `FlexCode` coefficients in parallel for further time savings. The computational complexity of `FlexCode` will be the same as n_{basis} parallel individual regressions. In particular, the scalability of `FlexCode` is determined by the underlying regression method. A scalable method like XGBoost leads to scalable `FlexCode` fitting. In the `Python` version of the code, the user can choose between the following regression methods: XGBoost for `FlexZBoost` but also nearest neighbors, LASSO (Tibshirani, 1996), and random forests regression (Breiman, 2001). In the `R` version, the following choices are available: XGBoost, nearest neighbors, LASSO, random forests, Nadaraya–Watson kernel smoothing (Nadaraya, 1964; Watson, 1964), sparse additive models (Ravikumar et al., 2009), and spectral series ("manifold") regression (Lee and Izbicki, 2016). In both implementations, the user may also use a custom regression method; we illustrate how this can be done with a vignette in both packages. For the `Python` implementation, the user can add any custom regression method following the `sklearn` API, i.e., with `fit` and `predict` methods. Examples in both languages are presented in Appendix A.3.

2.4. *DeepCDE*

Recently, neural networks have reemerged as a powerful tool for prediction settings where large amounts of representative training data are available; see LeCun et al. (2015) and Goodfellow et al. (2016) for a full review. Neural networks for CDE, such as Mixture Density Networks (MDNs; Bishop 1994) and variational methods (Tang and Salakhutdinov, 2013; Sohn et al., 2015), usually assume a Gaussian or some other parametric form of the conditional density. MDNs have lately also been used for photometric redshift estimation (D'Isanto and Polsterer, 2018; Pasquet et al., 2019) and for direct estimation of likelihoods and posteriors in cosmological parameter inference (see Alsing et al., 2019 and references within).

`DeepCDE`[7] (Dalmasso and Pospisil, 2019) takes a different, *fully nonparametric* approach to CDE. It combines the advantages of basis expansions with the flexibility of neural network architectures, allowing for data types like image features and time-series data. `DeepCDE` is based on the orthogonal series representation in `FlexCode`, given in Eq. (2), but rather than relying on regression methods to estimate the expansion coefficients in Eq. (2), `DeepCDE` computes the coefficients $\{\beta_i(\mathbf{x})\}_{i=1}^{B}$ *jointly* with a neural network that minimizes the CDE loss in Eq. (4). Indeed, one can show that for an orthogonal basis, the problem of minimizing this CDE loss is (asymptotically) equivalent to finding the best basis coefficients in `FlexCode` under mean squared error loss for the individual regressions; see Appendix C for a proof. The value of this result is that `DeepCDE` with a CDE loss directly connects prediction with uncertainty quantification, implying that one can leverage the state-of-the-art deep architectures for an application at hand toward uncertainty quantification for the same prediction setting.

From a neural network architecture perspective, `DeepCDE` only adds a linear output layer of coefficients for a series expansion of the density according to

$$\widehat{p}(y|\mathbf{x}) = \sum_{j=1}^{B} \widehat{\beta}_j(\mathbf{x})\phi_j(y), \tag{3}$$

where $\{\phi_j(y)\}_{j=1}^{B}$ is an orthogonal basis for functions of $y \in \mathbb{R}$. Like `FlexCode`, we normalize and remove spurious bumps from the final density estimates according to the procedure in Section 2.2 of Izbicki and Lee (2016).

The greatest benefit of `DeepCDE` is that it can be implemented with both convolutional and recurrent neural network architectures, extending to both image and sequential data. For most deep architectures, adding a linear layer represents a small modification, and a negligible increase in the number of parameters. For instance, with the AlexNet architecture (Krizhevsky et al., 2012), a widely used, relatively shallow convolutional neural network, adding a final layer with 30 coefficients for a cosine basis only adds ∼120,000 extra parameters. This represents a 0.1% increase over the 62 million already existing parameters, and hence a negligible increase in training and prediction time. Moreover, the CDE loss for `DeepCDE` is especially easy to evaluate; see Appendix C for details.

We provide both `TensorFlow` and `Pytorch` implementations of `DeepCDE` (Dalmasso and Pospisil, 2019). We also include examples that shows how one can easily build `DeepCDE` on top of AlexNet (Krizhevsky et al., 2012); in this case, for the task of estimating the probability distribution $\mathrm{p}(y|\mathbf{x})$ of the correct orientation y of a color image $\mathbf{x}$. Note that Fischer et al. (2015) use AlexNet for the corresponding regression task of predicting the orientation y for a color image $\mathbf{x}$ but without quantifying the uncertainty in the predictions.

3. How to assess method performance

After fitting CDEs, it is important to assess the quality of our models of uncertainty. For instance, after computing photo-z PDF estimates $\widehat{p}(z|\mathbf{x})$ for some galaxies, one may ask whether these estimates accurately quantify the true uncertainty distributions $\mathrm{p}(z|\mathbf{x})$. Similarly, in the LFI task, a key question is whether an estimate of the posterior distribution, $\widehat{p}(\boldsymbol{\theta}|\mathbf{x}_{\mathrm{obs}})$ of the cosmological parameters is close enough to the true posterior $\mathrm{p}(\boldsymbol{\theta}|\mathbf{x}_{\mathrm{obs}})$ given the observations $\mathbf{x}_{\mathrm{obs}}$.

We present two method-assessment tools suitable to different situations, which are complementary and can be performed simultaneously, with public implementations in the `cdetools`[8] package in both `Python` and `R` (Dalmasso et al., 2019). First, we describe a CDE loss function that directly provides relative comparisons between conditional density estimators, such as the methods presented in Section 2 or, equivalently, between a set of

[7] https://github.com/tpospisi/DeepCDE.

[8] https://github.com/tpospisi/cdetools.

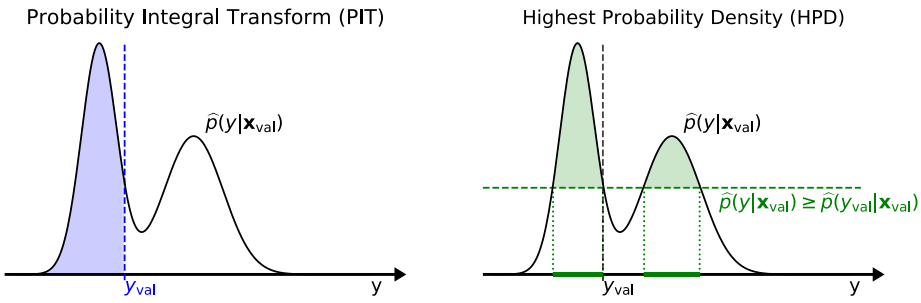

Fig. 2. Schematic diagram of the construction of the Probability Integral Transform (PIT, left) and the Highest Probability Density (HPD, right) values for the estimated density $\widehat{p}(y|\mathbf{x})$ at $\mathbf{x} = \mathbf{x}_{\text{val}}$, where y_{val} is the response at $x = \mathbf{x}_{\text{val}}$. In the plot to the right, the highlighted segments on the y-axis form the so-called highest density region (HDR) of $y|\mathbf{x}_{\text{val}}$. The PIT and HPD values correspond to the area of the tail versus highest density region, respectively, of the estimate; here indicated by the shaded areas. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

models (for the same method) with different tuning parameters. Second, we describe visual diagnostic tools, such as Probability Integral Transforms (PIT) and Highest Probability Density (HPD) plots, that can provide insights on the overall goodness-of-fit of a given estimator to observed data.

3.1. CDE loss

Here we briefly review the CDE loss from Izbicki and Lee (2016) for assessing conditional density estimators and discuss it in the context of the cosmology LFI case.

The goal of a loss function is to provide relative comparisons between different estimators, so that it is easy to directly choose the best fitted model among a list of candidates. Given an estimate $\widehat{p}$ of p, we define the CDE loss by

$$L(\widehat{p}, p) = \iint [\widehat{p}(\mathbf{y}|\mathbf{x}) - p(\mathbf{y}|\mathbf{x})]^2 \, d\mathbf{y} dP(\mathbf{x}), \tag{4}$$

where $P(\mathbf{x})$ is the marginal distribution of the features $\mathbf{x}$. This loss is the CDE analog to the standard mean squared error (MSE) in standard regression. The weighting by the marginal distribution of the features emphasizes that errors in the estimation of $\mathbf{y}$ for unlikely features $\mathbf{x}$ are less important. The CDE loss cannot be directly evaluated because it depends on the unknown true density $p(z|\mathbf{x})$. However, one can estimate the loss (up to a constant determined by the true p) by

$$\widehat{L}(\widehat{p}, p) = \frac{1}{n} \sum_{i=1}^{n} \int \widehat{p}(\mathbf{y}|\mathbf{x}_i^{te})^2 d\mathbf{y} - \frac{2}{n} \sum_{i=1}^{n} \widehat{p}(\mathbf{y}_i^{te}|\mathbf{x}_i^{te}), \tag{5}$$

where $\left\{(\mathbf{x}_i^{te}, \mathbf{y}_i^{te})\right\}_{i=1}^{n}$ represents our validation or test data, i.e., a held-out set not used to construct $\widehat{p}$. In our implementation, the function `cde_loss` returns the estimated CDE loss as well as an estimate of the standard deviation or the *standard error* (SE) of the estimated loss.

CDE loss for LFI. In LFI settings, we use a slightly different version of the CDE loss in Eq. (4). Because the goal (in ABC) is to approximate the posterior density $p(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$ at fixed $\mathbf{x} = \mathbf{x}_{\text{obs}}$, a natural evaluation metric is the integrated squared error loss

$$\int [\widehat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}) - p(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})]^2 \, d\boldsymbol{\theta} \tag{6}$$

of the conditional density *at* $\mathbf{x}_{obs}$ *only*. Estimating this loss can however be tricky as only a single instance of data with $x = \mathbf{x}_{\text{obs}}$ is available in practice. Hence, Izbicki et al. (2019) approximates Eq. (6) by computing the empirical loss $\widehat{L}(\widehat{p}, p)$ in Eq. (5) over a restricted subset of the validation data that only includes the $\mathbf{x}_i^{te}$ points that fall in an ϵ-neighborhood of $\mathbf{x}_{\text{obs}}$, where ϵ is the tolerance of the ABC rejection algorithm. The detailed analysis of this approach can be found in Izbicki et al. (2019).

3.2. PIT and HPD diagnostics

The CDE loss function is a relative measure of performance that cannot address absolute goodness-of-fit. To quantify overall goodness-of-fit, we examine how well calibrated an ensemble of conditional density estimators are on average, over validation or test data $\left\{(\mathbf{x}_i^{te}, \mathbf{y}_i^{te})\right\}_{i=1}^{n}$. For ease of notation, we will in this section denote $\mathbf{x}_i^{te}$ and $\mathbf{y}_i^{te}$ for a generic i by $\mathbf{x}_{val}$ and $\mathbf{y}_{val}$.

Given a true probability density $p(\mathbf{y}|\mathbf{x}) = \gamma_0$ of a variable $\mathbf{y}$ conditioned on data $\mathbf{x}$, an estimated probability density $\widehat{p}(\mathbf{y}|\mathbf{x}) = \gamma$ cannot be *well-calibrated* unless $\gamma \approx \gamma_0$.

Built on the same logic, the probability integral transform (PIT; Polsterer et al. 2016)

$$PIT(\mathbf{x}_{\text{val}}, y_{\text{val}}) = \int_{-\infty}^{y_{\text{val}}} \widehat{p}(y|\mathbf{x}_{\text{val}}) dy \tag{7}$$

assesses the calibration quality of an ensemble of CDEs for scalar y representing the cumulative distribution function (CDF) of $\widehat{p}(y|\mathbf{x}_{\text{val}})$ evaluated at $y = y_{\text{val}}$; this PIT value corresponds to the shaded area in Fig. 2, left. A statistically self-consistent population of densities has a uniform distribution of PIT values, and deviations from uniformity indicate inaccuracies of the estimated PDFs. Overly broad CDEs manifest as under-representation of the lowest and highest PIT values, whereas overly narrow CDEs manifest as over-representation of the lowest and highest values.

However, PIT values do not easily generalize to multiple responses. For instance, for a bivariate response $\mathbf{y} = (z, \eta)$, the quantity $\int_{-\infty}^{z_{\text{val}}} \int_{-\infty}^{\eta_{\text{val}}} \text{p}(z, \eta|\mathbf{x}_{\text{val}}) dz d\eta$ is not in general uniformly distributed (Genest and Rivest, 2001). An alternative statistic that easily generalizes to multivariate $\mathbf{y}$ is the highest probability density value (HPD; Izbicki et al. 2017, Appendix A):

$$\xi(\mathbf{x}_{\text{val}}, \mathbf{y}_{\text{val}}) = \int_{\mathbf{y}: \widehat{p}(\mathbf{y}|\mathbf{x}_{\text{val}}) \geq \widehat{p}(\mathbf{y}_{\text{val}}|\mathbf{x}_{\text{val}})} \widehat{p}(\mathbf{y}|\mathbf{x}_{\text{val}}) d\mathbf{y}. \tag{8}$$

The HPD value is based on the definition of the *highest density region* (HDR, Hyndman 1996) of a random variable $\mathbf{y}$; that is, the subset of the sample space of $\mathbf{y}$ where all points in the region have a probability above a certain value. The HDR of $\mathbf{y}|\mathbf{x}_{\text{val}}$ can be seen as a region estimate of $\mathbf{y}$ when $\mathbf{x} = \mathbf{x}_{\text{val}}$ is observed. In words, the set $\{\mathbf{y} : \widehat{p}(\mathbf{y}|\mathbf{x}_{\text{val}}) \geq \widehat{p}(\mathbf{y}_{\text{val}}|\mathbf{x}_{\text{val}})\}$ is the smallest HDR containing the point $\mathbf{y}_{\text{val}}$, and the HPD value is simply the probability of such a region. Fig. 2, right, shows a schematic diagram of the HPD value (green shaded area) and HDR region (highlighted segments on the y-axis) for the estimated density $\widehat{p}(\mathbf{y}|\mathbf{x}_{\text{val}})$. The HPD value $\xi(\mathbf{x}_{\text{obs}}, \mathbf{y}_{\text{val}})$ can also be viewed as a measure of how plausible $\mathbf{y}_{\text{val}}$ is according to $\widehat{p}(\mathbf{y}|\mathbf{x}_{\text{val}})$ and is directly related to the Bayesian analog of p-values or *the e-value* (Pereira and Stern, 1999). One can show (Harrison et al., 2015) that even for multivariate $\mathbf{y}$, the HPD values for validation data follow a

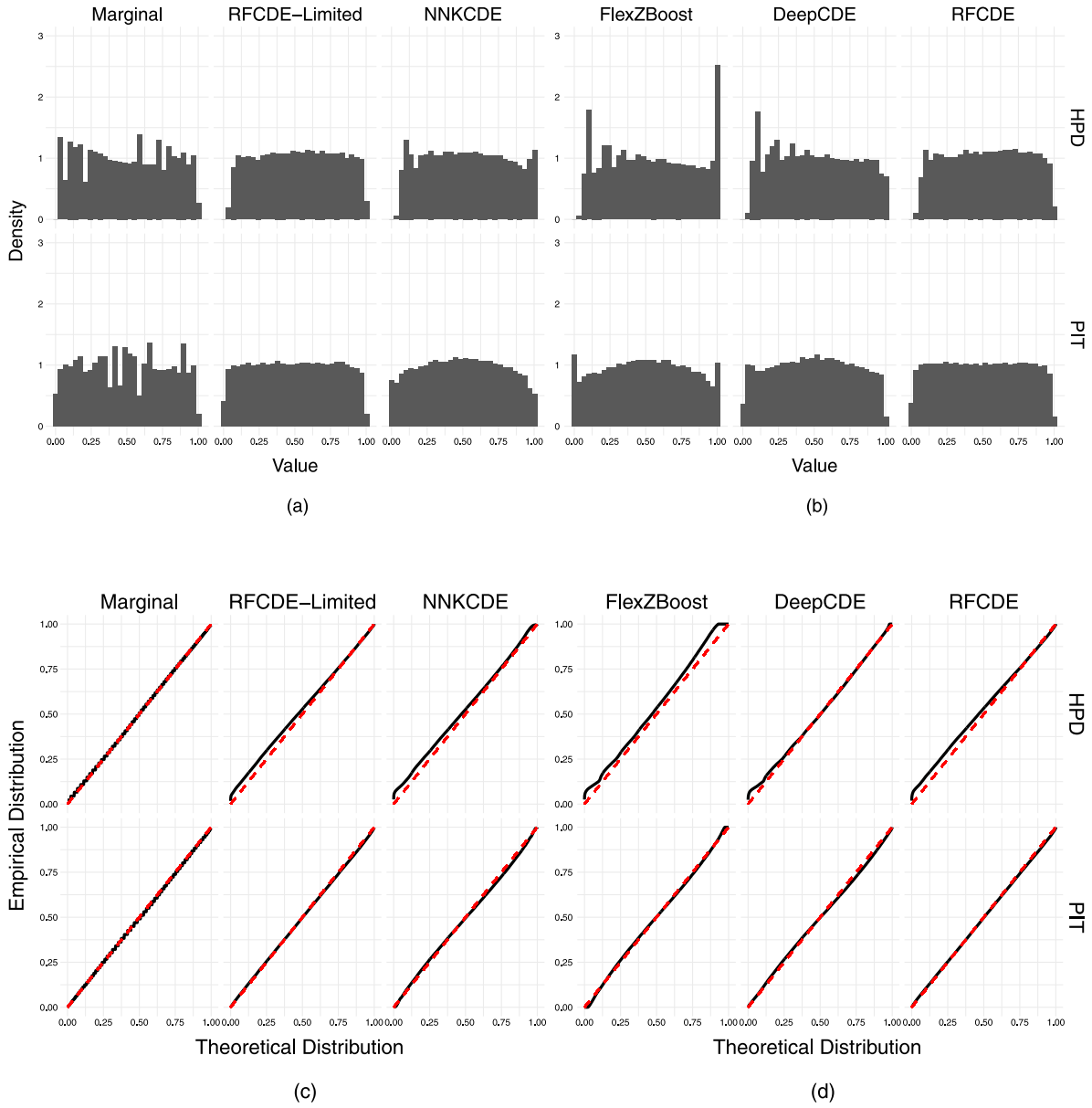

Fig. 3. (a)–(b) Histograms of PIT and HPD values for Teddy photo-z data. Both versions of RFCDE as well as the marginal distribution have a uniform distribution indicating that the conditional density estimates are well-calibrated on average. `FlexZBoost` exhibits an overrepresentation of 0 and 1 values which indicate overly narrow CDEs. (c)–(d) Probability–Probability (P–P) plots of HPD and PIT values. Both sets of values are computed over data in the test set and their empirical distribution is plotted against the uniform $U(0, 1)$ distribution; i.e., the "theoretical distribution" of the HPD and PIT values when $\widehat{p}(z|\mathbf{x}) = p(z|\mathbf{x})$. If the estimated CDEs are well calibrated, the empirical and theoretical distributions should coincide and all points should be close to the identity line. As in the top panel, the P–P plots indicate a good fit for all methods including the clearly misspecified "Marginal" model.

$U(0, 1)$ distribution if the CDEs are well calibrated on the population level. Thus, these values can also be used for assessing the fit of conditional densities in the same way as PIT values. In our implementation the functions `pit_coverage(cde_test, y_grid, y_test)` and `hpd_coverage(cde_test, y_grid, y_test)`, respectively, calculate PIT and HPD values for CDE estimates `cde_test` using the grid `y_grid` with observed values `y_test`.

The PIT and HPD are not without their limitations, however, as demonstrated in the control case of Schmidt et al. (2020) and Fig. 3 of Section 4.1 here. Because the PIT and HPD values can be uniformly distributed even if $\mathrm{p}(\mathbf{y}|\mathbf{x})$ is not well estimated, they must be used in conjunction with loss functions for method assessment. A popular way of visualizing PIT and HPD diagnostics for the entire population is through *probability–probability plots* or P–P plots of the empirical distribution of the (PIT or HPD) statistic versus its distribution under the hypothesis that $\widehat{p}(\mathbf{y}|\mathbf{x}) = p(\mathbf{y}|\mathbf{x})$; henceforth, we will refer to the latter Uniform(0,1) distribution as the "theoretical" distribution of PIT or HPD. An ideal P–P plot has all points close to the identity line where the "empirical" and "theoretical" distributions are the same. Note that HPD P–P plots, in particular, are valuable calibration tools if our goal is to calibrate the estimated densities so that the computed predictive regions have the right coverage.

Fig. 3(c)–(d) contains example P–P plots for the PIT and HPD values of photo-z estimators, where the empirical (observed) distribution is close to the theoretical (ideal) distribution.

4. Applications in astronomy

We demonstrate[9] the breadth of our CDE methods in three different astronomical use cases.

1. **Photo-z estimation: univariate response with multivariate input.** This is the standard prediction setting in which all four methods apply. In Section 4.1, we apply the methods to the TEDDY photometric redshift data (Beck et al., 2017), and illustrate the need for loss functions to properly assess PDF estimates of redshift z given photometric colors $\mathbf{x}$.
2. **Likelihood-free cosmological inference: multivariate response.** For multiple response components we want to model the often complicated dependencies between these components; this is in contrast to approaches which model each component separately, implicitly introducing an assumption of conditional independence. In Section 4.2, we use an example of LFI for simulated weak lensing shear data to show how `NNKCDE` and `RFCDE` can capture more challenging bivariate distributions with curved structures; in this toy example $\mathbf{y}$ represents cosmological parameters (Ω_M, σ_8) in the ΛCDM-model, and $\mathbf{x}$ represents (coarsely binned) weak lensing shear correlation functions.
3. **Spec-z estimation: functional input.** Standard prediction methods, such as random forests, do not typically fare well with functional features, simply treating them as unordered vectorial data and ignoring the functional structure. However, often there are substantial benefits to explicitly taking advantage of that structure as in `fRFCDE`. In Section 4.3, we compare the performance of a vectorial implementation of `RFCDE` with `fRFCDE` and `FlexCodeSeries` for a spectroscopic sample from the Sloan Digital Sky Survey (SDSS; Alam et al. 2015). The input $\mathbf{x}$ is here a high-resolution spectrum of a galaxy, and the response is the galaxy's redshift z.

Throughout this section we will report CDE loss mean and standard error for each method, using 95% Gaussian confidence intervals for performance comparison. Since the CDE loss is an empirical mean and the test size is reasonably large in all the examples, the validity of this approximation is guaranteed by the central limit theorem.

4.1. Photo-z estimation: Univariate response with multivariate input

Here we estimate photo-z posterior PDFs $\mathrm{p}(z|\mathbf{x})$ using representative training and test data (Samples A and B, respectively) from the TEDDY catalog by Beck et al. (2017).[10] These data include 74 309 training and 74 557 test observations. Each observation $\mathbf{x}$ has five features: the magnitude of the r-band and the pairwise differences or "colors" $u-g$, $g-r$, $r-i$, and $i-z$.

Among the chief sources of uncertainty affecting photo-zs estimated by ML techniques are the incompleteness and non-representativity of training sets, defined by the mismatch in the distributions of training and test data in z and $\mathbf{x}$, which may be extreme to the point of not guaranteeing mutual coverage. Realistically modeling incompleteness is highly challenging, requiring both simulations of SEDs and of the observational conditions of a given survey, which is outside the scope of this work. Accounting for redshift incompleteness is not a lost cause and may be accomplished by extrapolating outside of the training range by abandoning standard instance-based ML algorithms (see e.g., Leistedt and Hogg, 2016). Certain types of selection bias, known as covariate shift, can also be corrected by importance weights in the CDE loss (Izbicki et al., 2017; Freeman et al., 2017); see Appendix D for details and code. However, for simplicity, in this paper we consider only representative training sets with no disparity in color-space coverage, putting this demonstration on equal footing with all previous comparisons of photo-z PDF methods.

We fit `NNKCDE`, `RFCDE`, `DeepCDE`, and `FlexZBoost` to the data. In addition we fit an `RFCDE` model, "`RFCDE-Limited`", restricted to the first three of the five features, as a toy model which fails to extract some information from the features, allowing us to showcase the difference between PIT or HPD diagnostics and the CDE loss function. For `DeepCDE` we use a three-layer deep neural network with linear layers and reLu activations with 25 neurons per layers, trained for 10,000 epochs with Adam (Kingma and Ba, 2014). As our goal is to showcase its applicability we do not optimize the neural architecture (e.g., number of layers, number of neurons per layer, activation functions) nor the learning parameters (i.e., number of epochs, learning rate, momentum). To illustrate our validation methods, we also include the marginal distribution $\widehat{p}(z) = \frac{1}{n}\sum_{i=1}^{n}\widehat{p}(z|\mathbf{x}_i)$ as an estimate of individual photo-z distributions $p(z|\mathbf{x})$. This estimate will be the same regardless of $\mathbf{x}$.

Table 2 and Fig. 4 present the estimated CDE loss of Eq. (5) with estimated standard error (SE), as well as the storage space for each galaxy's photo-z CDE. Fig. 3 shows that the different models, including the clearly misspecified "Marginal" model, achieve comparable performance on goodness-of-fit diagnostics. However, Table 2 and Fig. 4 show the discriminatory power of the CDE loss function, which distinguishes the methods from one another. This emphasizes the need for method comparison through loss functions in addition to goodness-of-fit diagnostics. We note that the ranking in CDE loss also correlates roughly with the quality of the point estimates, as shown in Appendix B. The LSST-DESC PZ DC1 paper (Schmidt et al., 2020) draws similar conclusions from a comprehensive photo-z code comparison where the "Marginal" model (there referred to as `trainZ` photo-z PDF estimator, the experimental control) outperformed all codes when using traditional metrics for assessing photo-z PDF accuracy. Indeed, of the metrics considered in DC1, the CDE loss was the only metric that could appropriately penalize the pathological `trainZ`.

Table 2
Method comparison via the CDE loss of Eq. (5) with estimated standard error (SE) and the storage space for each galaxy's photo-z CDE. As `FlexZBoost` and `DeepCDE` are basis expansion methods, we need only to store the estimated coefficients for a lossless compression of the CDEs; the other CDEs are discretized to 200 bins.

Method	CDE loss $\pm$ SE	Storage (single CDE)
Marginal	-3.192 ± 0.007	200 floats (1.6 KB)
RFCDE-Limited	-11.038 ± 0.047	200 floats (1.6 KB)
NNKCDE	-12.139 ± 0.043	200 floats (1.6 KB)
FlexZBoost	-12.272 ± 0.044	30 coeffs (0.24 KB)
DeepCDE	-12.821 ± 0.04	31 coeffs (0.25 KB)
RFCDE	-13.108 ± 0.052	200 floats (1.6 KB)

4.2. Likelihood-free cosmological inference: Multivariate response

To showcase the ability to target joint distributions, we apply `NNKCDE` and `RFCDE` to the problem of estimating multivariate posteriors $p(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$ of the cosmological parameters in a likelihood-free setting via ABC–CDE (Izbicki et al., 2019).

ABC is an approach to parameter inference in settings where the likelihood is not tractable but we have a forward model that

[9] Code for these examples is publicly available at https://github.com/Mr8ND/cdetools_applications.

[10] Data available at https://github.com/COINtoolbox/photoz_catalogues.

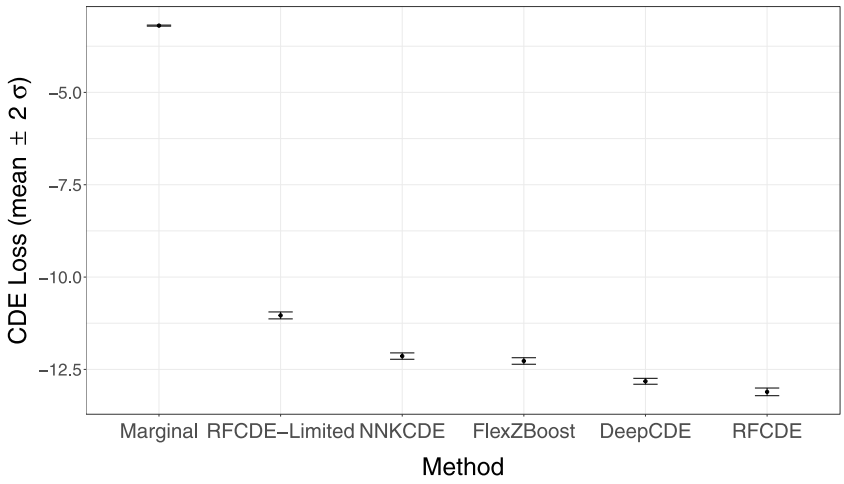

Fig. 4. Method comparison for Teddy photo-z data using CDE loss from Eq. (5). Dots provide the mean loss, while the upper and lower error bars correspond to ± 2 standard deviations.

can simulate data $\mathbf{x}$ under fixed parameter settings $\boldsymbol{\theta}$. The simplest form of ABC is the ABC rejection sampling algorithm, where a set of parameters is first drawn from a prior distribution. The simulated data $\mathbf{x}$ is accepted with tolerance $\varepsilon \geq 0$ if $d(\mathbf{x}, \mathbf{x}_{\text{obs}}) \leq \varepsilon$ for some distance metric d (e.g., the Euclidean distance) that measures the discrepancy between the simulated data $\mathbf{x}$ and observed data $\mathbf{x}_{\text{obs}}$. The outcome of the ABC rejection algorithm for small enough ϵ is a sample of parameter values approximately distributed according to the desired posterior distribution $p(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$.

The basic idea of ABC–CDE is to improve the ABC estimate – and hence reduce the number of required simulations – by using the ABC sample/output as input to a CDE method tuned with the CDE loss restricted to a neighborhood of $\mathbf{x}_{\text{obs}}$ defined by the tolerance ϵ (Izbicki et al., 2019, Equation 3). Hence, in ABC–CDE, our CDE method can be seen as a post-adjustment method: it returns an estimate $\widehat{p}(\boldsymbol{\theta}|\mathbf{x})$ which we evaluate at the point $\mathbf{x} = \mathbf{x}_{\text{obs}}$ to obtain a more accurate approximation of $p(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$. This could also be beneficial in an active learning setting where the posterior distribution is used to identify relevant regions of the parameter space (e.g., Lueckmann et al., 2019; Papamakarios et al., 2019; Alsing et al., 2019).

In this example, we consider the problem of cosmological parameter inference via *cosmic shear*, caused by weak gravitational lensing inducing a distortion in images of distant galaxies. The size and direction of the distortion is directly related to the size and shape of the matter distribution along the line of sight, which varies across the universe. We use shear correlation functions to constrain the dark matter density Ω_M and matter power spectrum normalization σ_8 parameters of the ΛCDM cosmological model, which predicts the properties and evolution of the large scale structure of the matter distribution. For further background see Hoekstra and Jain (2008), Munshi et al. (2008) and Mandelbaum (2018). Here we use the `GalSim`[11] toolkit (Rowe et al., 2015) to generate simplified galaxy shears distributed according to a Gaussian random field determined by (Ω_M, σ_8). The binned shear correlation functions serve as our input data or summary statistics x. For the inference, we assume uniform priors $\Omega_M \sim U(0.1, 0.8)$ and $\sigma_8 \sim U(0.5, 1.0)$ and fix $h = 0.7$, $\Omega_b = 0.045$, $z = 0.7$.

The top row of Fig. 5 shows the estimated bivariate posterior distribution of $\boldsymbol{\theta} = (\Omega_M, \sigma_8)$ from ABC rejection sampling only, at varying acceptance rates (20%, 50%, and 100%). An acceptance rate of 100% just returns the (uniform) ABC prior distribution, whereas the ABC posteriors for smaller acceptance rates (that is, smaller values of ε) concentrate around the parameter degeneracy curve (shown as a dashed line) on which the data are indistinguishable. The second and third rows show the estimated posteriors when we, respectively, improve the initial ABC estimate by applying `NNKCDE` and `RFCDE` tuned with our CDE surrogate loss. A result that is apparent from the figure is that `NNKCDE` and `RFCDE` fitted with a CDE loss are able to capture the degeneracy curve at a larger acceptance rate, that is, for a smaller number of simulations, than when using ABC only.

Table 3
Performance of ABC, `NNKCDE`, `RFCDE` in LFI settings with simulated weak lensing data in terms of the surrogate CDE loss.

Method \ acceptance rate	CDE loss $\pm$ SE ($\times 10^{-5}$)		
	20%	50%	100%
ABC	-0.686 ± 0.009	-0.392 ± 0.004	-0.227 ± 0.001
`NNKCDE`	-1.652 ± 0.022	-1.199 ± 0.016	-0.844 ± 0.010
`RFCDE`	-4.129 ± 0.063	-3.698 ± 0.064	-2.817 ± 0.055

We also have a direct measure of performance via the CDE loss and can adapt to different types of data by leveraging different CDE codes. Table 3 shows how the methods compare in terms of the surrogate loss. While decreasing the acceptance rate benefits all methods, it is clear that CDE-based approaches have better performance in all cases.

4.3. Spec-z estimation: Functional input

In this example, we compare CDE methods in the context of spectroscopic redshift prediction using 2812 high-resolution spectra from the Sloan Digital Sky Survey (SDSS) Data Release 6 (preprocessed with the cuts of Richards et al. 2009), corresponding to features $\mathbf{x}$ of flux measurements at $d = 3501$ wavelengths. The high-resolution spectra $\mathbf{x}$ can be seen as functional inputs on a continuum of wavelength values. Spectroscopic redshifts (or redshifts z predicted from spectra $\mathbf{x}$) tend to be both accurate and precise, so the density $p(z|\mathbf{x})$ is well-approximated by a delta function at the true redshift. For the purposes of illustrating the use of the CDE codes, we define a noisified redshift $z_i = z_i^{\text{SDSS}} + \epsilon_i$, where ϵ_i are independent and identically distributed variables drawn from a normal distribution $N(0, 0.02)$ and z_i^{SDSS} is the true redshift of galaxy i provided by SDSS. Thus the conditional density

[11] https://github.com/GalSim-developers/GalSim.

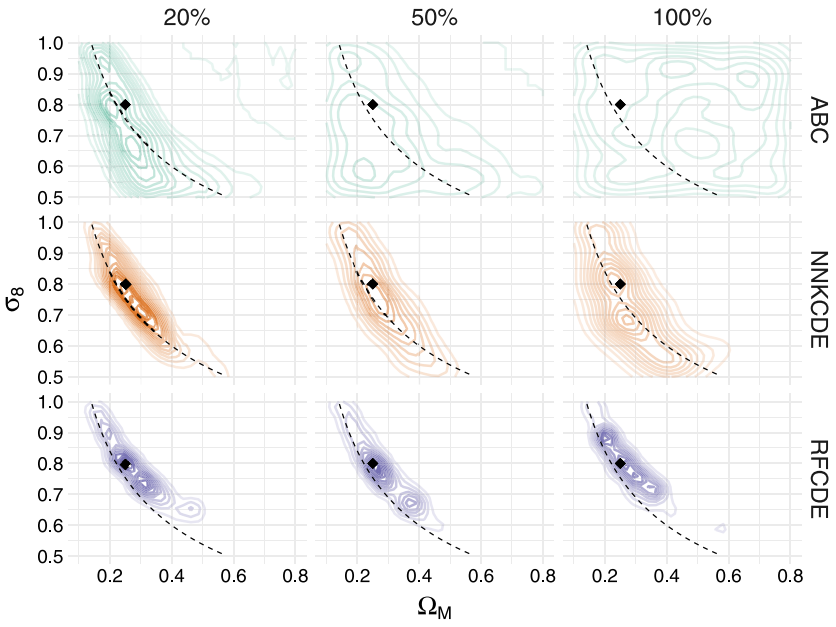

Fig. 5. Cosmological parameter inference in an LFI setting with simulated weak lensing data. The top row shows the estimated bivariate distribution of Ω_M and σ_8 for ABC rejection sampling at different acceptance rates (20%, 50%, and 100%). The middle row shows the estimated posterior densities after applying `NNKCDE` to the ABC sample, and the bottom row when applying `RFCDE`. Both `NNKCDE` and `RFCDE` tuned with a CDE loss improve on ABC across acceptance rate levels, i.e., they provide approximate posteriors that are more concentrated around the true observed parameter values and the degeneracy curve on which the data are indistinguishable (shown here as a black diamond and dashed line, respectively). We even see some structure at an ABC acceptance rate of 1 (right column); that is, at an ABC threshold of $\epsilon \to \infty$ for which the entire sample is accepted by ABC and passed to our CDE code. HPD values were not included as they fail to distinguish between conditional and marginal distribution, as mentioned above.

Table 4
Performance of `RFCDE`for functional data. We achieve both a lower CDE loss and computational time for Functional `RFCDE`(as compared to Vector `RFCDE`) by leveraging the functional nature of the data.

Method	Train time (in s)	CDE loss ± SE
Functional `RFCDE`	24.89	−3.38 ± 0.155
Vector `RFCDE`	41.60	−2.52 ± 0.104
Regression RF + KDE	50.63	−3.42 ± 0.120
`FlexCode-Spec`	4017.93	−3.53 ± 0.136

$p(z_i|\mathbf{x}_i)$ of this example is a Gaussian distribution with mean z_i^{SDSS} and variance 0.02.

We compare `fRFCDE`, a "Functional" adaptation of `RFCDE`, with a standard "Vector" implementation of `RFCDE`, which treats functional data as a vector. For completeness, we also compare against standard regression random forest combined with KDE, as well as `FlexCode-Spec`, an extension of `FlexCode` with a Spectral Series regression (Richards et al., 2009; Freeman et al., 2009; Lee and Izbicki, 2016) for determining the expansion coefficients. We train on 2000 galaxies and test on the remaining galaxies. The `RFCDE` and the `fRFCDE` trees are both trained with $n_{trees} = 1000$, $n_{basis} = 31$, and bandwidths chosen by plug-in estimators. We use $\lambda = 50$ for the `fRFCDE` rate parameter of the Poisson process that defines the variable groupings.

Table 4 contains the CDE loss and train time for both the vector-based and functional-based `RFCDE` models on the SDSS data, as well as for `FlexCode-Spec`. We obtain substantial gains when incorporating functional features both in terms of CDE loss and computational time. The computational gains are attributed to requiring fewer searches for each split point as the default value of $m_{try} = \sqrt{d}$ is reduced. As anticipated, vector-based `RFCDE` underperforms with functional data due to the tree splits on CDE loss struggling to pick up signals in the data and returning almost the same conditional density regardless of the input. In contrast, splitting on mean squared error is an easier task and we include the results of regular regression random forest and KDE, which are comparable to the results of `fRFCDE`. Whereas random forests rely on variable selection (either individual variables as in vectorial `RFCDE` or grouped together as in `fRFCDE`) for dimensional reduction, `FlexCode-Spec` is based on the Spectral Series mechanism for dimension reduction that finds (potentially) nonlinear, sparse structure in the data distribution (Izbicki and Lee, 2016). Both types of dimension reduction are reasonable for spec-z estimation, as particular wavelength locations or regions in the galaxy SED could carry information of the galaxy's true redshift; `RFCDE` and `fRFCDE` are effective in finding such locations by variable selection. Spectral Series on the other hand are able to recover low-dimensional manifold structure in the entire data ensemble; as Richards et al. 2009 show, the main direction of variation in SDSS spectra is directly related to the spectroscopic redshift.

5. Conclusions

This paper presents statistical tools and software for uncertainty quantification in complex regression and parameter inference tasks. Given a set of features $\mathbf{x}$ with associated response $\mathbf{y}$, our methods extend the usual point prediction task of classification and regression to nonparametric estimation of the entire (conditional) probability density $p(\mathbf{y}|\mathbf{x})$. The described CDE methods are meant to handle a range of different data settings as outlined in Table 1. This paper includes examples of code usage in the contexts of photo-z estimation, likelihood-free inference for cosmology analysis, and spec-z estimation. In addition, it provides tools for CDE method assessment and for choosing tuning parameters of the CDE methods in a principled way.

Our software includes four packages for CDE — `NNKCDE`, `RFCDE`, `FlexCode` and `DeepCDE`— each using a different machine learning algorithm, as well as a package for model assessment, `cdetools`. All packages are implemented in both `Python` and `R`, with no dependence on proprietary software, making them compatible with any operating system supporting either language

(e.g., Windows, MacOS, Ubuntu, and any other Unix-based OS). The code is provided in publicly available Github repositories, documented using `Python` and `R` function documentation standards, and equipped with `Travis CI`[12] automatic bug-tracking. Finally, our software makes uncertainty quantification straightforward for those used to standard open-source machine learning `Python` packages: `NNKCDE`, `RFCDE`, `FlexCode` share the `sklearn` API (with `fit` and `predict` methods), which makes our methods compatible with `sklearn` wrapper functions (for e.g., cross-validation and model ensembling). `DeepCDE` has implementations for both `Tensorflow` and `Pytorch`, two of the most widely used deep learning frameworks, and can easily be combined with most existing network architectures.

CRediT authorship contribution statement

N. Dalmasso: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review and editing, Visualization. **T. Pospisil:** Conceptualization, Methodology, Software, Validation, Writing – original draft. **A.B. Lee:** Conceptualization, Methodology, Writing – original draft, Writing – review and editing, Project administration, Funding acquisition, Supervision. **R. Izbicki:** Conceptualization, Methodology, Software, Writing – original draft, Writing – review and editing, Supervision. **P.E. Freeman:** Writing – original draft. **A.I. Malz:** Writing – original draft, Writing – review and editing.

Acknowledgments

We would like to thank the two anonymous reviewers for their insightful comments that helped improve the manuscript. ND, TP and ABL were partially supported by the National Science Foundation under Grant No. DMS1521786. RI was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (grant number 306943/2017-4) and Fundação de Amparo à Pesquisa do Estado de São Paulo (grant numbers 2017/03363-8 and 2019/11321-9). AIM acknowledges support from the Max Planck Society and the Alexander von Humboldt Foundation in the framework of the Max Planck-Humboldt Research Award endowed by the Federal Ministry of Education and Research. During the completion of this work, AIM was advised by David W. Hogg and supported in part by National Science Foundation grant AST-1517237.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at doi:10.1016/j.ascom.2019.100362.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. http://tensorflow.org/.

Abbott, T.M.C., Alarcon, A., Allam, S., Andersen, P., Andrade-Oliveira, F., Annis, J., Asorey, J., Avila, S., Bacon, D., Banik, N., Bassett, B.A., Baxter, E., Bechtol, K., Becker, M.R., Bernstein, G.M., Bertin, E., Blazek, J., Bridle, S.L., Brooks, D., Brout, D., Burke, D.L., Calcino, J., Camacho, H., Campos, A., Carnero Rosell, A., Carollo, D., Carrasco Kind, M., Carretero, J., Castander, F.J., Cawthon, R., Challis, P., Chan, K.C., Chang, C., Childress, M., Crocce, M., Cunha, C.E., D'Andrea, C.B., da Costa, L.N., Davis, C., Davis, T.M., De Vicente, J., DePoy, D.L., DeRose, J., Desai, S., Diehl, H.T., Dietrich, J.P., Dodelson, S., Doel, P., Drlica-Wagner, A., Eifler, T.F., Elvin-Poole, J., Estrada, J., Evrard, A.E., Fernandez, E., Flaugher, B., Foley, R.J., Fosalba, P., Frieman, J., Galbany, L., García-Bellido, J., Gatti, M., Gaztanaga, E., Gerdes, D.W., Giannantonio, T., Glazebrook, K., Goldstein, D.A., Gruen, D., Gruendl, R.A., Gschwend, J., Gutierrez, G., Hartley, W.G., Hinton, S.R., Hollowood, D.L., Honscheid, K., Hoormann, J.K., Hoyle, B., Huterer, D., Jain, B., James, D.J., Jarvis, M., Jeltema, T., Kasai, E., Kent, S., Kessler, R., Kim, A.G., Kokron, N., Krause, E., Kron, R., Kuehn, K., Kuropatkin, N., Lahav, O., Lasker, J., Lemos, P., Lewis, G.F., Li, T.S., Lidman, C., Lima, M., Lin, H., Macaulay, E., MacCrann, N., Maia, M.A.G., March, M., Marriner, J., Marshall, J.L., Martini, P., McMahon, R.G., Melchior, P., Menanteau, F., Miquel, R., Mohr, J.J., Morganson, E., Muir, J., Möller, A., Neilsen, E., Nichol, R.C., Nord, B., Ogando, R.L.C., Palmese, A., Pan, Y.C., Peiris, H.V., Percival, W.J., Plazas, A.A., Porredon, A., Prat, J., Romer, A.K., Roodman, A., Rosenfeld, R., Ross, A.J., Rykoff, E.S., Samuroff, S., Sánchez, C., Sanchez, E., Scarpine, V., Schindler, R., Schubnell, M., Scolnic, D., Secco, L.F., Serrano, S., Sevilla-Noarbe, I., Sharp, R., Sheldon, E., Smith, M., Soares-Santos, M., Sobreira, F., Sommer, N.E., Swann, E., Swanson, M.E.C., Tarle, G., Thomas, D., Thomas, R.C., Troxel, M.A., Tucker, B.E., Uddin, S.A., Vielzeuf, P., Walker, A.R., Wang, M., Weaverdyck, N., Wechsler, R.H., Weller, J., Yanny, B., Zhang, B., Zhang, Y., Zuntz, J., DES Collaboration, 2019. Cosmological constraints from multiple probes in the dark energy survey. Phys. Rev. Lett. 122, 171301. doi:10.1103/PhysRevLett.122.171301, URL: https://link.aps.org/doi/10.1103/PhysRevLett.122.171301.

Aghanim, N., Akrami, Y., Ashdown, M., Aumont, J., Baccigalupi, C., Ballardini, M., Banday, A., Barreiro, R., Bartolo, N., Basak, S., et al., 2018. Planck 2018 results. VI. Cosmological parameters. arXiv preprint arXiv:1807.06209.

Alam, S., Albareti, F.D., Prieto, C.A., Anders, F., Anderson, S.F., Anderton, T., Andrews, B.H., Armengaud, E., Aubourg, É., Bailey, S., et al., 2015. The eleventh and twelfth data releases of the Sloan Digital Sky Survey: final data from SDSS-III. Astrophys. J. Suppl. Ser. 219 (1), 12.

Alsing, J., Charnock, T., Feeney, S., Wandelt, B., 2019. Fast likelihood-free cosmology with neural density estimators and active learning. Mon. Not. R. Astron. Soc. 488 (3), 4440–4458. doi:10.1093/mnras/stz1960.

Alsing, J., Wandelt, B., Feeney, S., 2018. Massive optimal data compression and density estimation for scalable, likelihood-free inference in cosmology. Mon. Not. R. Astron. Soc. 477 (3), 2874–2885. doi:10.1093/mnras/sty819.

Ball, N.M., Brunner, R.J., 2010. Data mining and machine learning in astronomy. Internat. J. Modern Phys. D 19 (07), 1049–1106. doi:10.1142/S0218271810017160.

Beaumont, M.A., Zhang, W., Balding, D.J., 2002. Approximate Bayesian computation in population genetics. Genetics 162 (4), 2025–2035, https://www.genetics.org/content/162/4/2025.

Beck, R., Lin, C.-A., Ishida, E., Gieseke, F., de Souza, R., Costa-Duarte, M., Hattab, M., Krone-Martins, A., Collaboration, C., 2017. On the realistic validation of photometric redshifts. Mon. Not. R. Astron. Soc. 468 (4), 4323–4339. doi:10.1093/mnras/stx687.

Bishop, C.M., 1994. Mixture Density Networks. Technical Report, Citeseer.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32. doi:10.1023/A:1010933404324.

Carrasco Kind, M., Brunner, R.J., 2013. TPZ: photometric redshift PDFs and ancillary information by using prediction trees and random forests. Mon. Not. R. Astron. Soc. 432 (2), 1483–1501. doi:10.1093/mnras/stt574.

Carrasco Kind, M., Brunner, R.J., 2014. Sparse representation of photometric redshift probability density functions: preparing for petascale astronomy. Mon. Not. R. Astron. Soc. 441 (4), 3550–3561. doi:10.1093/mnras/stu827, URL: https://academic.oup.com/mnras/article/441/4/3550/1229381/Sparse-representation-of-photometric-redshift.

Chen, T., Guestrin, C., 2016. XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 785–794. doi:10.1145/2939672.2939785.

Chen, Y., Gutmann, M.U., 2019. Adaptive Gaussian copula ABC. In: Chaudhuri, K., Sugiyama, M. (Eds.), Proceedings of Machine Learning Research. In: Proceedings of Machine Learning Research, vol. 89, PMLR, pp. 1584–1592, URL: http://proceedings.mlr.press/v89/chen19d.html.

Cunha, C.E., Lima, M., Oyaizu, H., Frieman, J., Lin, H., 2009. Estimating the redshift distribution of photometric galaxy samples–II. Applications and tests of a new method. Mon. Not. R. Astron. Soc. 396 (4), 2379–2398. doi:10.1111/j.1365-2966.2009.14908.x.

Dalmasso, N., Pospisil, T., 2019. tpospisi/DeepCDE 0.1. doi:10.5281/zenodo.3364862.

Dalmasso, N., Pospisil, T., Izbicki, R., 2019. tpospisi/cdetools 0.0.1. doi:10.5281/zenodo.3364810.

D'Isanto, A., Polsterer, K., 2018. Photometric redshift estimation via deep learning-Generalized and pre-classification-less, image based, fully probabilistic redshifts. Astron. Astrophys. 609, A111. doi:10.1051/0004-6361/201731326.

Epanechnikov, V.A., 1969. Non-parametric estimation of a multivariate probability density. Theory Probab. Appl. 14 (1), 153–158. doi:10.1137/1114019.

[12] https://docs.travis-ci.com/.

Fischer, P., Dosovitskiy, A., Brox, T., 2015. Image orientation estimation with convolutional networks. In: GCPR. doi:10.1007/978-3-319-24947-6_30.

Frank, E., Hall, M., 2001. A simple approach to ordinal classification. In: De Raedt, L., Flach, P. (Eds.), Machine Learning: ECML 2001. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 145–156.

Freeman, P.E., Izbicki, R., Lee, A.B., 2017. A unified framework for constructing, tuning and assessing photometric redshift density estimates in a selection bias setting. Mon. Not. R. Astron. Soc. 468 (4), 4556–4565. doi:10.1093/mnras/stx764.

Freeman, P.E., Newman, J.A., Lee, A.B., Richards, J.W., Schafer, C.M., 2009. Photometric redshift estimation using Spectral Connectivity Analysis. Mon. Not. R. Astron. Soc. 398, 2012–2021.

Genest, C., Rivest, L.-P., 2001. On the multivariate probability integral transformation. Statist. Probab. Lett. 53 (4), 391–399.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Graff, P., Feroz, F., Hobson, M.P., Lasenby, A., 2014. SKYNET: an efficient and robust neural network training tool for machine learning in astronomy. Mon. Not. R. Astron. Soc. 441 (2), 1741–1759. doi:10.1093/mnras/stu642.

Greenberg, D.S., Nonnenmacher, M., Macke, J.H., 2019. Automatic posterior transformation for likelihood-free inference. In: Proceedings of the 36th International Conference on Machine Learning. ICML 2019, 9–15 June 2019. Long Beach, California, USA, pp. 2404–2414, URL: http://proceedings.mlr.press/v97/greenberg19a.html.

Harrison, D., Sutton, D., Carvalho, P., Hobson, M., 2015. Validation of Bayesian posterior distributions using a multidimensional Kolmogorov–Smirnov test. Mon. Not. R. Astron. Soc. 451 (3), 2610–2624.

Hoekstra, H., Jain, B., 2008. Weak gravitational lensing and its cosmological applications. Ann. Rev. Nucl. Part. Sci. 58, 99–123.

Hyndman, R.J., 1996. Computing and graphing highest density regions. Amer. Statist. 50 (2), 120–126.

Hyndman, R., Einbeck, J., Wand, M., 2018. hdrcde: Highest density regions and conditional density estimation. R package version 3.3. https://cran.r-project.org/web/packages/hdrcde/hdrcde.pdf.

Ivezić, Ž., Connolly, A.J., VanderPlas, J.T., Gray, A., 2014. Statistics, Data Mining, and Machine Learning in Astronomy: a Practical Python Guide for the Analysis of Survey Data, Vol. 1. Princeton University Press.

Ivezić, Ž., Kahn, S.M., Tyson, J.A., Abel, B., Acosta, E., Allsman, R., Alonso, D., AlSayyad, Y., Anderson, S.F., Andrew, J., 2019. LSST: From science drivers to reference design and anticipated data products. Astrophys. J. 873 (2), 111. doi:10.3847/1538-4357/ab042c.

Izbicki, R., Lee, A.B., 2016. Nonparametric conditional density estimation in a high-dimensional regression setting. J. Comput. Graph. Statist. 25 (4), 1297–1316. doi:10.1080/10618600.2015.1094393.

Izbicki, R., Lee, A.B., 2017. Converting high-dimensional regression to high-dimensional conditional density estimation. Electron. J. Stat. 11 (2), 2800–2831. doi:10.1214/17-EJS1302.

Izbicki, R., Lee, A.B., Freeman, P.E., 2017. Photo-z estimation: An example of nonparametric conditional density estimation under selection bias. Ann. Appl. Stat. 11 (2), 698–724. doi:10.1214/16-AOAS1013.

Izbicki, R., Lee, A.B., Pospisil, T., 2019. ABC–CDE: Toward approximate Bayesian computation with complex high-dimensional data and limited simulations. J. Comput. Graph. Statist. 1–20. doi:10.1080/10618600.2018.1546594.

Izbicki, R., Lee, A., Schafer, C., 2014. High-dimensional density ratio estimation with extensions to approximate likelihood computation. In: Artificial Intelligence and Statistics. pp. 420–429.

Izbicki, R., Pospisil, T., 2019. rizbicki/FlexCode v5.9-beta.3. doi:10.5281/zenodo.3366065.

Joudaki, S., Blake, C., Johnson, A., Amon, A., Asgari, M., Choi, A., Erben, T., Glazebrook, K., Harnois-Déraps, J., Heymans, C., Hildebrandt, H., Hoekstra, H., Klaes, D., Kuijken, K., Lidman, C., Mead, A., Miller, L., Parkinson, D., Poole, G.B., Schneider, P., Viola, M., Wolf, C., 2017. KiDS-450 + 2dFLenS: Cosmological parameter constraints from weak gravitational lensing tomography and overlapping redshift-space galaxy clustering. Mon. Not. R. Astron. Soc. 474 (4), 4894–4924. doi:10.1093/mnras/stx2820.

Juric, M., Axelrod, T., Becker, A.C., Becla, J., Bellm, E., Bosch, J.F., Ciardi, D., Connolly, A.J., Dubois-Felsmann, G.P., Economou, F., Freemon, M., Gelman, M., Graham, M., Ivezić, Ž., Jenness, T., Kantor, J., Krughoff, K., Lim, K.-T., Lupton, R.H., Mueller, F., Nidever, D., Patterson, M., Petravick, D., Shaw, D., Slater, C., Strauss, M., Swinbank, J., Tyson, J.A., Wood-Vasey, M., Wu, X., 2017. Data Products Definition Document. LSST Corporation, URL: https://docushare.lsstcorp.org/docushare/dsweb/Get/LSE-163/.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. CoRR abs/1412.6980.

Krause, E., Eifler, T., Zuntz, J., Friedrich, O., Troxel, M., Dodelson, S., Blazek, J., Secco, L., MacCrann, N., Baxter, E., et al., 2017. Dark energy survey year 1 results: Multi-probe methodology and simulated likelihood analyses. arXiv preprint arXiv:1706.09359.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems, Vol. 1. NIPS'12, Curran Associates Inc., USA, pp. 1097–1105, URL: http://dl.acm.org/citation.cfm?id=2999134.2999257.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521 (7553), 436. doi:10.1038/nature14539.

Lee, A.B., Izbicki, R., 2016. A spectral series approach to high-dimensional nonparametric regression. Electron. J. Stat. 10 (1), 423–463. doi:10.1214/16-EJS1112.

Leistedt, B., Hogg, D., 2016. Data-driven, interpretable photometric redshifts trained on heterogeneous and unrepresentative data. Astrophys. J. 838, doi:10.3847/1538-4357/aa6332.

Li, J., Nott, D., Fan, Y., Sisson, S., 2017. Extending approximate Bayesian computation methods to high dimensions via a Gaussian copula model. Comput. Statist. Data Anal. 106, 77–89. doi:10.1016/j.csda.2016.07.005.

Lueckmann, J.-M., Bassetto, G., Karaletsos, T., Macke, J.H., 2019. Likelihood-free inference with emulator networks. In: Symposium on Advances in Approximate Bayesian Inference. pp. 32–53.

Lueckmann, J.-M., Gonçalves, P.J., Bassetto, G., Öcal, K., Nonnenmacher, M., Macke, J.H., 2017. Flexible statistical inference for mechanistic models of neural dynamics. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17, Curran Associates Inc., USA, pp. 1289–1299, URL: http://dl.acm.org/citation.cfm?id=3294771.3294894.

Malz, A.I., Marshall, P.J., DeRose, J., Graham, M.L., Schmidt, S.J., Wechsler, R., L.D.E.S. Collaboration, 2018. Approximating photo- z PDFs for large surveys. Astron. J. 156 (1), 35. doi:10.3847/1538-3881/aac6b5.

Mandelbaum, R., 2018. Weak lensing for precision cosmology. Ann. Rev. Astron. Astrophys. 56 (1), 393–433. doi:10.1146/annurev-astro-081817-051928.

Mandelbaum, R., Seljak, U., Hirata, C.M., Bardelli, S., Bolzonella, M., Bongiorno, A., Carollo, M., Contini, T., Cunha, C.E., Garilli, B., 2008. Precision photometric redshift calibration for galaxy-galaxy weak lensing. Mon. Not. R. Astron. Soc. 386 (2), 781/806. doi:10.1111/1365-2966.2008.12947.

Marin, J.-M., Raynal, L., Pudlo, P., Ribatet, M., Robert, C., 2016. ABC random forests for Bayesian parameter inference. Bioinformatics 35, doi:10.1093/bioinformatics/bty867.

Meinshausen, N., 2006. Quantile regression forests. J. Mach. Learn. Res. 7, 983–999, URL: http://dl.acm.org/citation.cfm?id=1248547.1248582.

Munshi, D., Valageas, P., Van Waerbeke, L., Heavens, A., 2008. Cosmology with weak lensing surveys. Phys. Rep. 462 (3), 67–121.

Nadaraya, E.A., 1964. On estimating regression. Theory Probab. Appl. 9 (1), 141–142.

Ntampaka, M., Avestruz, C., Boada, S., Caldeira, J., Cisewski-Kehe, J., Di Stefano, R., Dvorkin, C., Evrard, A.E., Farahi, A., Finkbeiner, D., et al., 2019. The role of machine learning in the next decade of cosmology. arXiv preprint arXiv:1902.10159.

Papamakarios, G., Murray, I., 2016. Fast ε-free inference of simulation models with Bayesian conditional density estimation. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 29. Curran Associates, Inc., pp. 1028–1036.

Papamakarios, G., Sterratt, D., Murray, I., 2019. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In: The 22nd International Conference on Artificial Intelligence and Statistics. pp. 837–848.

Pasquet, J., Bertin, E., Treyer, M., Arnouts, S., Fouchez, D., 2019. Photometric redshifts from SDSS images using a convolutional neural network. Astron. Astrophys. 621, A26. doi:10.1016/j.ascom.2016.03.006.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. PyTorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. J. Mach. Learn. Res. 12, 2825–2830, URL: http://dl.acm.org/citation.cfm?id=1953048.2078195.

Pereira, C.A.d.B., Stern, J., 1999. Evidence and credibility: full Bayesian significance test for precise hypotheses. Entropy 1 (4), 99–110.

Polsterer, K.L., D'Isanto, A., Gieseke, F., 2016. Uncertain photometric redshifts. arXiv preprint arXiv:1608.08016.

Pospisil, T., Dalmasso, N., 2019a. tpospisi/NNKCDE 0.3. doi:10.5281/zenodo.3364858.

Pospisil, T., Dalmasso, N., 2019b. tpospisi/RFCDE 0.3.2. doi:10.5281/zenodo.3364856.

Pospisil, T., Dalmasso, N., Inacio, M., 2019. tpospisi/FlexCode 0.1.5. doi:10.5281/zenodo.3364860.

Pospisil, T., Lee, A.B., 2018. RFCDE: Random forests for conditional density estimation. arXiv preprint arXiv:1804.05753.

Pospisil, T., Lee, A.B., 2019. (f)RFCDE: Random forests for conditional density estimation and functional data. arXiv preprint: arXiv:1906.07177.

Ravikumar, P., Lafferty, J., Liu, H., Wasserman, L., 2009. Sparse additive models. J. R. Stat. Soc. Ser. B Stat. Methodol. 71 (5), 1009–1030. doi:10.1111/j.1467-9868.2009.00718.x, URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9868.2009.00718.x.

Richards, J.W., Freeman, P.E., Lee, A.B., Schafer, C.M., 2009. Exploiting low-dimensional structure in astronomical spectra. Agron. J. 691, 32–42.

Rowe, B., Jarvis, M., Mandelbaum, R., Bernstein, G.M., Bosch, J., Simet, M., Meyers, J.E., Kacprzak, T., Nakajima, R., Zuntz, J., et al., 2015. GALSIM: The modular galaxy image simulation toolkit. Astron. Comput. 10, 121–150. doi:10.1016/j.ascom.2015.02.002.

Schmidt, S., Malz, A., Soo, J., Almosallam, I., Brescia, M., Cavuoti, S., Cohen-Tanugi, J., Connolly, A., DeRose, J., Freeman, P., Graham, M., Iyer, K., Jarvis, M., Kalmbach, J., Kovacs, E., Lee, A., Longo, G., Morrison, C., LSST Dark Energy Science Collaboration, 2020. Evaluation of probabilistic photometric redshift estimation approaches for lsst. arXiv:2001.03621.

Sheldon, E.S., Cunha, C.E., Mandelbaum, R., Brinkmann, J., Weaver, B.A., 2012. Photometric redshift probability distributions for galaxies in the SDSS DR8. Astrophys. J. Suppl. Ser. 201 (2), 32. doi:10.1088/0067-0049/201/2/32.

Sisson, S.A., Fan, Y., Beaumont, M., 2018. Handbook of Approximate Bayesian Computation. Chapman and Hall/CRC, doi:10.1201/9781315117195.

Sohn, K., Lee, H., Yan, X., 2015. Learning structured output representation using deep conditional generative models. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 28. Curran Associates, Inc., pp. 3483–3491, URL: http://dl.acm.org/citation.cfm?id=2969442.2969628.

Tang, Y., Salakhutdinov, R.R., 2013. Learning stochastic feedforward neural networks. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 530–538, URL: http://dl.acm.org/citation.cfm?id=2999611.2999671.

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B Stat. Methodol. 58 (1), 267–288. doi:10.2307/41262671.

van Uitert, E., Joachimi, B., Joudaki, S., Amon, A., Heymans, C., Köhlinger, F., Asgari, M., Blake, C., Choi, A., Erben, T., Farrow, D.J., Harnois-Déraps, J., Hildebrandt, H., Hoekstra, H., Kitching, T.D., Klaes, D., Kuijken, K., Merten, J., Miller, L., Nakajima, R., Schneider, P., Valentijn, E., Viola, M., 2018. KiDS+GAMA: cosmology constraints from a joint analysis of cosmic shear, galaxy-galaxy lensing, and angular clustering. Mon. Not. R. Astron. Soc. 476 (4), 4662–4689. doi:10.1093/mnras/sty551.

Vanderplas, J., Connolly, A., Ivezić, Ž., Gray, A., 2012. Introduction to astroml: machine learning for astrophysics. In: Conference on Intelligent Data Understanding. CIDU, pp. 47–54. doi:10.1109/CIDU.2012.6382200.

Viironen, K., López-Sanjuan, C., Hernández-Monteagudo, C., Chaves-Montero, J., Ascaso, B., Bonoli, S., Cristóbal-Hornillos, D., Díaz-García, L.A., Fernández-Soto, A., Márquez, I., Masegosa, J., Pović, M., Varela, J., Cenarro, A.J., Aguerri, J.A.L., Alfaro, E., Aparicio-Villegas, T., Benítez, N., Broadhurst, T., Cabrera-Caño, J., Castander, F.J., Cepa, J., Cerviño, M., González Delgado, R.M., Husillos, C., Infante, L., Martínez, V.J., Moles, M., Molino, A., del Olmo, A., Perea, J., Prada, F., Quintana, J.M., 2018. High redshift galaxies in the ALHAMBRA survey. II. Strengthening the evidence of bright-end excess in UV luminosity functions at 2.5 $\leq z \leq$ 4.5 by PDF analysis. Astron. Astrophys. 614, A129. doi:10.1051/0004-6361/201731797, arXiv:1712.01028.

Watson, G.S., 1964. Smooth regression analysis. Sankhyā 359–372.

Way, M.J., Scargle, J.D., Ali, K.M., Srivastava, A.N., 2012. Advances in Machine Learning and Data Mining for Astronomy. Chapman and Hall/CRC.

Wittman, D., 2009. What lies beneath: Using p(z) to reduce systematic photometric redshift errors. Astrophys. J. Lett. 700 (2), L174. doi:10.1088/0004-637X/700/2/L174.