Multiresolution Scalable Bayesian Conditional Density Estimation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Nonparametric estimation of the conditional distribution of a response given high-dimensional features is a challenging problem. In many settings it is important to allow not only the mean but also the variance and shape of the response density to change flexibly with features, which are massive-dimensional with a distribution concentrated near a lower-dimensional subspace or manifold. We propose a multiresolution model based on a novel stick-breaking prior placed on the dictionary weights. The algorithm scales efficiently to massive numbers of features, and can be implemented efficiently with slice sampling. State of the art predictive performance is demonstrated for toy examples and a real data application.

1 Introduction

Massive datasets are becoming a ubiquitous by-product of modern scientific and industrial applications. These data present statistical and computational challenges for machine learning because many previously developed approaches do not scale-up sufficiently. Specifically, challenges arise because of the ultrahigh-dimensionality, and relatively low sample size (the "large p, small n" problem). Parsimonious models for such big data assume that the density in the ambient dimension concentrates around a lower-dimensional (possibly nonlinear) subspace. Indeed, a plethora of methodologies are emerging to estimate such lower-dimensional "manifolds" from high-dimensional data [1, 2].

We are interested in using such lower-dimensional embeddings to obtain estimates of the conditional distribution of some target variable(s). This *conditional regression* setting arises in a number of important application areas, including neuroscience, genetics, and video processing. For example, one might desire automated estimation of a predictive density for a continuous neurologic *phenotype* of interest, such as intelligence or a creativity score, on the basis of available data for a patient including neuroimaging. The challenge is to estimate the probability density function of the phenotype *nonparametrically* based on an $\mathcal{O}(10^6)$ dimensional image of the subject's brain. It is crucial to avoid parametric assumptions on the density, such as Gaussianity, while allowing the density to change flexibly with predictors. Otherwise, one can obtain misleading predictions and poorly characterize predictive uncertainty.

There is a rich machine learning and statistical literature on conditional density estimation of a response $y \in \mathcal{Y}$ given a set of features (predictors) $x = (x_1, x_2, \dots, x_p) \in \mathcal{X}$. Common approaches include hierarchical mixtures of experts [3, 4], kernel methods [5, 6, 7, 8], Bayesian finite mixture models [9, 10, 11] and Bayesian nonparametrics [12, 13, 14, 15, 16].

However, there has been limited consideration of scaling to large p settings, with the variational Bayes approach of [10] being a notable exception. For dimensionality reduction, Tran et al. follow a greedy variable selection algorithm. Their approach does not scale to the sized applications we are interested in. For example, in a problem with $p = 1,000$ and $n = 500$, they reported a CPU time of 51.7 minutes for a single analysis. We are interested in problems with p and n having many more orders of magnitude, requiring a faster computing time while also accommodating flexible non-linear dimensionality reduction (variable selection is a limited sort of dimension reduction). To our knowledge, there are no nonparametric density regression competitors to our approach, which maintain a characterization of uncertainty in estimating the conditional densities; rather, all sufficiently scalable algorithms provide point predictions and/or rely on restrictive assumptions such as linearity.

In big data problems, scaling is often accomplished using divide-and-conquer techniques. Well known examples are classification and regression trees (CART) [17] and multivariate adaptive regression splines (MARS) [18]. These algorithms fit surfaces to data by explicitly dividing the input space into a nested sequence of regions, and by fitting simple surfaces within these regions. Though these methods are appealing in providing a simple, flexible and interpretable mechanism of dimension reduction, it is well known that single tree estimates commonly have high variance and poor performance. There is a rich literature proposing improvements based on bagging [19], boosting [20] and random forests [21]. Though these algorithms can substantially improve mean square error performance, computation can be expensive and performance degrades as dimensionality p increases.

In fact, a significant downside of many divide-and-conquer algorithms is their poor scalability to high dimensional predictors. As the number of features increases, the problem of finding the best splitting attribute becomes intractable so that CART, MARS and multiple trees models cannot be efficiently applied. Also mixture of experts models become computationally demanding, since both mixture weights and dictionary densities are predictor dependent. In an attempt to make mixtures of experts more efficient, sparse extensions relying on different variable selection algorithms have been proposed [22]. However, performing variable selection in high dimensions is effectively intractable: algorithms need to efficiently search for the best subsets of predictors to include in weight and mean functions within a mixture model, an NP-hard problem.

In order to efficiently deal with massive datasets, we propose a novel multiresolution approach which starts by learning a multiscale dictionary of densities,. This tree is efficiently learned in a first stage using a fast and scalable graph partitioning algorithm applied to the high-dimensional features [23]. Expressing the conditional densities $f(y|x)$ for each $x \in \mathcal{X}$ as a convex combination of coarse-to-fine scale dictionary densities, the learning problem in the second stage estimates the corresponding multiresolution probability tree. This is accomplished in a Bayesian manner using a novel multiresolution stick-breaking process, which allows the data to inform about the optimal bias-variance tradeoff; weighting coarse scale dictionary densities more highly decreases variance while adding to bias if the finer scale structure is needed. This results in a model that allows borrowing information across different resolution levels and reaches a good compromise in terms of the bias-variance tradeoff. We show that the algorithm scales efficiently to massive numbers of features.

2 Setting

Let $X \colon \Omega \to \mathcal{X} \subseteq \Re^p$ be a p-dimensional Euclidean vector-valued predictor random variable, taking values $x \in \mathcal{X}$, with a marginal probability distribution F_X. Similarly, let $Y \colon \Omega \to \mathcal{Y} \subseteq \Re$ be a scalar-valued target random variable, taking values $y \in \mathcal{Y}$, with a marginal probability distribution F_Y. Our goal is to develop an approach that facilitates obtaining an estimate of $F_{Y|X}$ given n pairs of observations that we assume are sampled exchangeable from the joint distribution, $(x_i, y_i) \sim F_{X,Y}$. Let $\mathcal{D}_n = \{(x_i, y_i)\}_{i \in [n]}$, where $[n] = \{1, \ldots, n\}$.

Our approach is indirect. Rather than directly estimating $F_{Y|X}$, we posit the existence of a latent random variable $Z \colon \Omega \to \mathcal{Z} \subseteq \mathcal{X}$, where $\mathcal{Z}$ is only d "dimensional", where $d \ll p$. Note that $\mathcal{Z}$ need not be a linear subspace of $\mathcal{X}$, rather, $\mathcal{Z}$ could be, for example, a union or affine subspaces, or a smooth compact Riemannian manifold. Regardless of the nature of $\mathcal{Z}$, we assume that we can approximately decompose the joint distribution as follows, $F_{X,Y,Z} = F_{X,Y|Z} F_Z \approx F_{X|Z} F_{Y|Z} F_Z$. In other words, we assume that the *conditional signal* approximately concentrates around a low-dimensional latent space. Note that this is a much less restrictive assumption than the commonplace assumption in manifold learning that the marginal distribution, F_X concentrates around a low-dimensional latent space.

To provide some intuition around this model, we provide the following concrete example. Let $Z \sim U(0,1)$, and let F_X be constructed as follows. $x_1 = z\sin(z)$, $x_2 = z\cos(z)$, and $X_j \sim \mathcal{N}(0,1)$ for all $j \in \{3,\ldots,50\}$. Moreover, let $Y \sim \mathcal{N}(Z, Z+1)$. Thus, clearly, Y is conditionally dependent on Z, which is the low-dimensional signal manifold, of which X is also a function. In particular, X lives on a swissroll embedded in a 50-dimensional ambient space, but Y is only a function of where Z is along the swissroll. Figure **??** depicts this concrete example.

3 General Methodology

We propose here a general modular methodology consisting of four components: (i) a tree decomposition of the space, (ii) an embedding $\psi \colon \mathcal{X} \to \Psi$ of the x_i's that we assume the y_i's are dependent upon, (iii) an assumed form of the conditional probability model, $\mathcal{P}_{Y|\psi(X)}$, and (iv) a prior over scales, π.

Tree Decomposition A tree decomposition yields a multiscale partition of the data. Let $(\mathcal{W}, \rho_W, F_W)$ be a measurable metric space, where $F_W = F_{X,Y}$ is a Borel probability measure, $\mathcal{W} = \mathcal{X} \times \mathcal{Y}$, and $\rho_W \colon \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ is a metric on $\mathcal{W}$. Let $B_r^{\mathcal{V}}(v)$ be the ρ_W-ball inside $\mathcal{V}$ of radius $r > 0$ centered at $w \in \mathcal{V}$. We define a tree decomposition as in [**?**]:

Definition 1 *A* tree decomposition τ *of a* d*-dimensional metric measure space* $(\mathcal{V}, \rho_V, F_V)$ *is a collection of open sets* $\{\mathcal{C}_{j,k}\}_{j\in\mathcal{K}_k, k\in\mathbb{Z}}$*, called* cells*, satisfying:*

(i) for all $j \in \mathbb{Z}$*,* $F_V(\mathcal{V} \setminus \cup_{k\in\mathcal{K}_j} C_{j,k}) = 0$*,*

(ii) for all $j' \geq j$ *and* $k' \in \mathcal{K}_j$*, either* $C_{j',k'} \subseteq C_{j,k}$ *or* $F_V(C_{j',k\cap C_{j,k}}) = 0$*,*

(iii) for $j < j'$ *and* $k' \in \mathcal{K}_{j'}$*, there exists a unique* $k \in \mathcal{K}_j$ *such that* $C_{j',k'} \subseteq C_{j,k}$*,*

(iv) each $C_{j,k}$ *contains a point* $c_{j,k}$ *such that* $B^{\mathcal{V}}_{r\cdot 2^{-j}}(c_{j,k}) \subseteq C_{j,k} \subseteq B^{\mathcal{V}}_{2^{-j}}(c_{j,k})$ *for a constant* r *depending on the intrinsic geometric properties of* $\mathcal{V}$*. In particular, we have* $F_V(C_{j,k}) \approx 2^{-dj}$.

The first condition means that at each scale, the set of cells covers the space almost everywhere. The second and third condition together mean that as we descend the tree, going from a coarser to finer scale, for a given cell at scale j*, it is the child of a single coarser scale cell, such that its intersection with any other coarser scale cell has measure zero. The fourth condition means that there exists a point* $c_{j,k}$ *which is effectively the location of cell* $C_{j,k}$*, in that it is in* $C_{j,k}$*, and that all other points in* $C_{j,k}$ *are within a small distance of* $c_{j,k}$*.*

Given a tree decomposition, we can approximate $F_{Y|X}$ at each scale j by $\cup_{k\in\mathcal{K}_j} F_{Y|C_{j,k}}$.

4 Model specification

Our approach proceeds in a two stage fashion as follows. We first learn a multiscale nonlinear partition tree of the feature data $\{x_i\}_{i=1}^n$, recursively partitioning $\{x_i\}$ to obtain subsets that are increasingly homogeneous according to some metric. The coarsest scale contains all of the data, the next finer scale contains two or more clusters of observations having relatively similar x_i values, and so on until our convergence criteria are met (for example, the available sample size is exhausted so that few observations fall within each fine scale leaf partition set). Based on the multiscale partition, each value of x has an associated path through the tree encoding the set membership at each scale. Using the response data y_i for all subjects i in a given partition set, we estimate a dictionary density specific to that set and resolution level using Bayesian methods. The conditional density is then expressed as a convex combination of these multiresolution dictionary densities, with a posterior distribution on the weights learned under a multiresolution stick-breaking process.

Definition 2 *A tree decomposition of a* p *dimensional space* $\mathcal{X}$ *is a collections of sets* $\{\mathcal{X}_j^m\}_{j\in\mathcal{K}_m, m\in\mathcal{S}}$ *satisfying:*

(i) for every $m \in \mathcal{S}$, $\cup_{j=1}^{\mathcal{K}_j} \mathcal{X}_j^m = \mathcal{X}$

(ii) for every $j \neq j'$ and $m \in \mathcal{S}$, $\mathcal{X}_j^m \cap \mathcal{X}_{j'}^m = \emptyset$

(iii) for every j and $m > 1$ there is a j' such that $\mathcal{X}_m^j \subset \mathcal{X}_{m-1}^{j'}$

For simplicity we will focus on dyadic trees where each set is split into two subsets. Starting from generation one corresponding to the entire $\mathcal{X}$, denoted as $\mathcal{X}^1$, each set $\mathcal{X}_j^m$ is split into two mutually exclusive partition sets so that for a general partition level m the partition will be given by $\mathcal{X}^m = \left(\mathcal{X}_1^m, \ldots, \mathcal{X}_{2^{m-1}}^m\right)$. Let us assume this process proceeds for L levels and that all final leaves contain one element. Let (m, s) be the node associated to the sth subset at resolution level m. Let $A_m(x) \in \{1, \ldots, 2^{m-1}\}$ be the location of predictor x at level m, with $A_1(x)$ equal to 1 by definition. The vector $A(x) = [A_1(x), \ldots, A_L(x)]$ encodes the path through the partition tree up to generation L specific to predictor value x. Let $de(\ell, s)$ and $an(\ell, s)$ be respectively the set of descendants and ancestors of node (ℓ, s).

We characterize the conditional density $f(y|x)$ as a convex combination of multiscale dictionary densities. At level one, the global parent density is denoted by f_1. For predictor value x, the dictionary density at generation j is $f_{B_j(x)}$ with $B_j(x) = \{j, A_j(x)\}$, for $j = 2, \ldots, k$. Then, $f(y|x)$ is defined as the convex combination of densities $\{f_{B_j(x)}\}_{j=1}^k$ with weights $\{\pi_{B_j(x)}\}_{j=1}^k$, i.e.

$$f(y|x) = \sum_{j=1}^{k} \pi_{B_j(x)} f_{B_j(x)}(y), \qquad (1)$$

where $0 \leq \pi_{B_j(x)}$ and $\sum_{j=1}^k \pi_{B_j(x)} = 1$.

Each $B(x)$ is a set encoding the path through the partition tree up to generation k specific to predictor value x. According to model (1), one observation can lie in subsets located at different resolution levels. This is critical in achieving a good compromise between bias and variance through borrowing information across different resolution levels. Though the proposed approach is reminiscent of a mixture of experts model [3], the two approaches are quite different, since under (1), neither mixture weights nor dictionary densities directly depend on predictors. This allows our model to scale efficiently to high dimensional predictors.

Now let us examine the implications of model (1). For two predictor values x and x' located close together, it is expected that the paths will be similar, which leads to similar weights on the dictionary densities. In the extreme case in which x and x' belong to the same leaf partition set, we have $B(x) = B(x')$ and the path through the tree will be the same. Moreover, in this case, we will have $f(y|x) = f(y|x')$ so that up to k levels of resolution the densities $f(y|x)$ and $f(y|x')$ are identical. If the paths through the tree differ only in the final generation or two, the weights will typically be similar but the resulting conditional densities will not be identical.

To derive mixture weights, a natural choice corresponds to a stick-breaking process [24]. For each node $B_j(x_i)$ in the partition tree, define a stick length $V\{B_j(x_i)\} \sim \text{Beta}(1, \alpha)$. The parameter α encodes the complexity of the model, with $\alpha = 0$ corresponding to the case in which $f(y|x) = f(y)$. We relate the weights in (1) to the stick-breaking random variables as follows:

$$\pi_{B_j(x)} = V\left\{B_j(x)\right\} \prod_{B_h \in an\{B_j\}} \left[1 - V\left\{B_h(x)\right\}\right],$$

with $V\{B_k(x)\} = 1$ to ensure that $\sum_{j=1}^k \pi_{B_j}(x) = 1$. We refer to this prior as a *multiresolution stick-breaking process*.

5 Estimation

The proposed approach is based on a two-stage algorithm where first the observations are allocated to different subsets in a tree fashion using an efficient partitioning algorithm and then, considering the partition as fixed, a multiresolution stick-breaking process is estimated. In practice, observations are partitioned applying metis [23], a fast multiscale technique used for graph partitioning. Though more complicated densities can be considered, dictionary densities f_{B_j} will be estimated by assuming a normal form, i.e. $f_{B_j} = \mathcal{N}(\eta_{B_j}, \sigma_{B_j})$. In particular, densities corresponding to a particular partition set will be estimated considering only observations belonging to that partition set. To be specific, for estimating density $f_{B_j}(y)$, we use the data $\{y_i : x_i \in \mathcal{X}^j_{A_j}\}$. We then conduct the analysis treating partition sets as fixed; this is critical for scalability to big p. On the surface, conditioning on a single tree seems overly restrictive, but using a second stage multiresolution probability model for the weights over the tree leads to inferences that are robust to the tree estimate; almost as if the tree itself were randomized.

Parameters involved in the dictionary densities can be estimated using either frequentist or Bayesian methods. Bayesian methods are appealing since they can avoid singularities associated with traditional maximum likelihood inference, the prior has an appealing role as a regularizer, and we can characterize uncertainty in dictionary learning through the resulting posterior. Hence, parameters involved in dictionary densities will be estimated through Bayesian methods and inference on stick breaking weights and dictionary density parameters will be carried out using the Gibbs sampler. For this purpose, introduce the latent variable $S_i \in \{1, \dots, k\}$, for $i = 1, \dots, n$, denoting the multiscale level used by the ith subject. Assuming data are normalized prior to analysis, we let $\eta \sim \mathcal{N}(0, I)$ and $\sigma = \mathcal{IG}(a, b)$ for the means and variances of the dictionary densities. Let n_{B_j} be the number of observations allocated to node B_j . Each Gibbs sampler iteration can be summarized in the following steps.

(i) Update S_i by sampling from the multinomial full conditional with

$$\Pr(S_i = j \mid -) = \frac{\pi_{B_j(x_i)} f_{B_j(x_i)}(y_i)}{\sum_{h=1}^{k} \pi_{B_h(x_i)} f_{B_h(x_i)}(y_i)}$$

(ii) Update stick-breaking random variable $V_{B_j(x_i)}$, for $j = 1, \dots, k$ and $i = 1, \dots, n$, from Beta(β_p, α_p) with $\beta_p = 1 + n_{B_j}$ and $\alpha_p = \alpha + \sum_{B_h(x_i) \in de\{B_j(x_i)\}} n_{B_h(x_i)}$.

(iii) Update $(\eta_{B_j(x_i)}, \sigma_{B_j(x_i)})$ by sampling from

$$\eta_{B_j} \sim \mathcal{N}\left(\bar{y}_{B_j} n_{B_j}/\sigma_{B_j}, (1 + n_{B_j}/\sigma_{B_j})^{-1}\right)$$

$$\sigma_{B_j} \sim \mathcal{IG}\left(a_\sigma, b + 0.5 \sum_{\{i: S_i = j, x_i \in B_j\}} \left(y_i - \eta_{B_j}\right)^2\right)$$

with $a_\sigma = a + n_{B_j}/2$, $\bar{y}_{B_j}$ being the average of the observation $\{y_i\}$ allocated to node B_j.

6 Simulation studies

In order to assess the predictive performance of the proposed model, different simulation scenarios were considered. Let n be the number of observations, $y \in \Re$ the response variable and $x \in \Re^p$ a set of predictors. The Gibbs sampler was run considering $20,000$ as the maximum number of iterations with a burn-in of $1,000$. Gibbs sampler chains were stopped testing normality of normalized averages of functions of the Markov chain [25]. Parameters (a, b) and α involved in the prior density of parameters σ_{B_j}s and V_{B_j}s were set respectively equal to $(3, 1)$ and 1.

In all simulation scenarios, predictors were assumed to belong to an r-dimensional space, either a lower dimensional plane or a non linear manifold, with $r << p$. For each synthetic dataset, the proposed model was compared with CART and lasso in terms of mean squared error. For CART and Lasso standard Matlab packages were utilized. In order to fairly compare Lasso with the proposed model, a fast Lasso algorithm based on Lars was implemented and the regularization parameter was chosen based on the AIC.

6.1 Illustrative Example

First let us consider the simple toy example of §2. We created an equally spaced grid of points $t_i = 0, \dots, 20$. Then, we let $\eta_i = \sin(t_i)$ and predictors be a linear function of η_i plus Gaussian noise, i.e. $x_i = \eta_i + \epsilon_i$ with $\epsilon_i \sim N(0, 0.1)$. The response was drawn from the following mixture of Gaussians

$$y_i \sim w_i \mathcal{N}(-2, 1) + (1 - w_i)\mathcal{N}(2, 1) \tag{2}$$

with $w_i = |\eta_i|$. Our model was run considering different sample sizes. Figure 1 shows the estimated density of two data points. These estimates were obtained by performing leave-one-out prediction for different number of observations in the training set. As the figure clearly shows our construction facilitates an estimate of the density y that become closer to the true density as the number of observations in the training set increases.

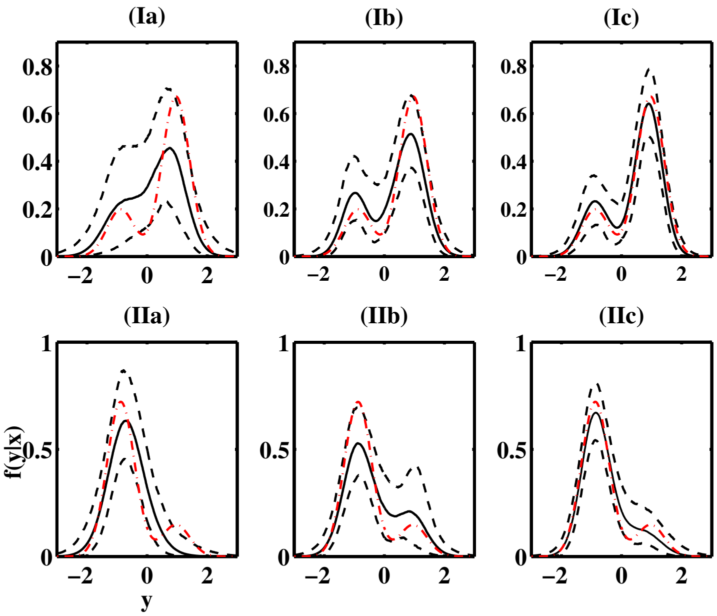

Figure 1: Illustrative example: Plot of true (red dashed-dotted line) and estimated (50th percentile: solid line, 2.5th and 97.5th percentiles: dashed lines) density for two data points (I, II) considering different training set size (a:50, b:100, c:150).

6.2 Linear lower dimensional space

In this section, the vector of predictors was assumed to lie close to a lower dimensional plane. In practice, predictors were modeled through a factor model as follows

$$x_i = \Lambda \eta_i + \epsilon_i \tag{3}$$

with $\epsilon_i \sim \mathcal{N}(0, \Sigma_0)$, $\Sigma_0 = diag(\sigma_1, \dots, \sigma_p)$, Λ being a $p \times r$ matrix, $\eta_i \sim \mathcal{N}(0, I)$ and $r << p$. In the first simulation scenario the response y was assumed to be a function of the latent variable η so that the dependence between response and predictors was induced by the shared dependence on the latent factors. In practice, the pair (y_i, x_i) was jointly sampled from a factor model. The loading matrix was derived as the product of a matrix with orthogonal columns and a diagonal matrix with positive elements on the diagonal, i.e. $\Lambda = \Gamma\Theta$. In particular, the columns of Γ were uniformly sampled from the Stiefel manifold while the diagonal matrix of Θ were sampled from an inverse Gamma with shape and rate parameters $(1, 4)$. In the second simulation scenario, x was sampled from a factor model with sparse loading while y was sampled from a normal with location and scale parameter $(1, 1)$ if the first variable was positive, i.e. $x_1 > 0$, and from a normal with location and scale $(-1, 1)$ otherwise. In this example, the non zero elements of the loading matrix were sampled from a normal with zero mean and standard deviation 3. In all the examples, an inverse gamma prior with parameters $(1, 4)$ were utilized for σ_j with $j = 1, \dots, p$.

Table 1 and 2 show mean squared errors under the proposed approach, CART and lasso based on leave-one-out prediction. As shown in table 1 and table 2, in almost all data scenario, our model is

Table 1: Linear manifold example 1: Mean and standard deviations of squared errors under multi-scale stick-breaking (MSB), CART and Lasso for sample size 50 and 100 for different simulation scenarios.

				$r = 5$			$r = 10$	
p	n		MSB	CART	LASSO	MSB	CART	LASSO
$10e + 03$	50	MSE	0.18	0.31	0.25	0.22	0.58	0.22
		STD	0.32	0.30	0.42	0.24	0.54	0.30
		TIME	3	2	1	3	3	1
$10e + 03$	100	MSE	0.18	0.27	0.26	0.20	0.41	0.52
		STD	0.26	0.42	0.46	0.23	0.46	0.78
		TIME	5	5	2	5	5	1
$10e + 04$	50	MSE	0.35	0.45	0.89	0.16	0.33	0.20
		STD	0.53	0.77	1.04	0.21	0.46	0.31
		TIME	3	25	2	3	27	2
$10e + 04$	100	MSE	0.43	0.88	0.52	0.17	0.50	0.31
		STD	0.59	1.29	0.70	0.24	0.75	0.49
		TIME	7	50	5	7	51	5
$50e + 04$	50	MSE	0.11	0.16	0.15	0.83	2.26	0.92
		STD	0.15	0.24	0.19	1.01	2.60	3.69
		TIME	5	90	11	5	121	10
$50e + 04$	100	MSE	0.003	0.17	0.08	0.13	1.37	1.06
		STD	0.16	0.23	0.13	1.12	1.81	1.50
		TIME	10	214	43	8	227	42
$70e + 04$	50	MSE	1.70	1.48	1.47	0.66	1.65	1.07
		STD	2.18	2.47	1.63	0.87	1.49	0.95
		TIME	6	121	12	7	151	13
$50e + 04$	100	MSE	0.69	1.36	0.82	0.78	1.52	1.43
		STD	0.94	1.47	1.28	1.03	1.34	2.11
		TIME	13	321	41	12	325	44

able to perform as well as or better than the model associated to the lowest mean squared error. In the first data scenario, given the linear relationship between response and predictors, Lasso performs better CART in almost all experiments. On the other hand, the non linear relationship between response and predictors assumed in the second data scenario results in better performance for CART. Table 1 and figure 3 show the mean of CPU usage to predict a single point as a function of the number of features. In particular, CPU time is expressed in seconds and codes have been running on our workstation (Intel Core i7-2600K Quad-Core Processor memory 8192 MB). Clearly, the proposed model scale substantially better than others to high dimensional predictors.

6.3 Non-Linear lower dimensional space

In this section predictors were assumed to lie close to a lower dimensional non-linear manifold. In the first simulation study, predictors and response were jointly sampled from an N components mixture of factor analyzers so that the vector of predictors and response were assumed to lie close to N lower dimensional planes. For each mixture components, the loading matrix and variances were sampled as in the first simulation scenario in §6.2, while mixture weights were sampled from a Dirichlet distribution with parameter $\alpha_j = 1$ for $j = 1, \ldots, N$. The number of latent factors was considered to be increasing in the number of components, in practice we let the hth mixture component be modeled through h factors. In the other simulation scenarios predictors were assumed to lie close to the Swissroll and the S-manifold (see figure 2), all two dimensional manifold embedded in

Table 2: Linear manifold example 2: Mean and standard deviations of squared errors under multiscale stick-breaking (MSB), CART and Lasso for sample size 50 and 100

				$r=2$			$r=5$	
p	n		MSB	CART	LASSO	MSB	CART	LASSO
$10e+03$	100	MSE	1.54	1.78	2.37	0.84	1.25	1.62
		STD	1.70	1.72	0.89	1.38	1.35	1.47
$50e+03$	100	MSE	0.76	0.97	1.77	0.88	1.53	1.43
		STD	1.04	1.21	3.13	1.00	1.59	2.73
$10e+04$	100	MSE	0.77	1.01	1.61	0.67	0.46	0.97
		STD	0.94	1.13	1.85	0.82	0.61	1.16
$20e+04$	100	MSE	0.86	0.90	1.41	0.74	1.09	0.78
		STD	1.30	1.35	1.41	0.95	1.98	0.95

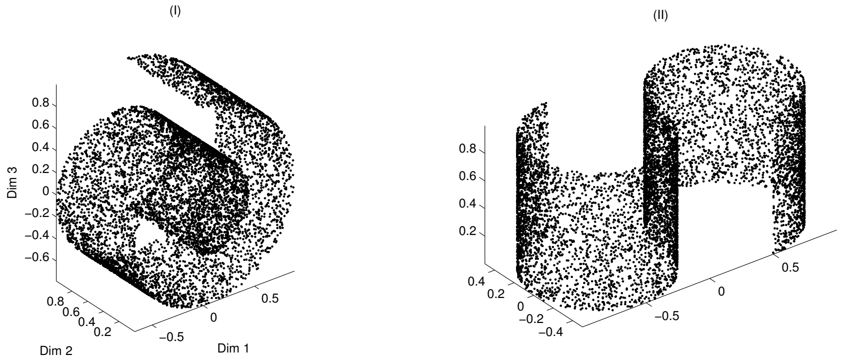

Figure 2: Non-linear manifolds: Swissroll (I) and S-Manifold (II) embedded in $\mathcal{R}^3$

$\Re^p$ while the response was sampled from a normal with mean equal to one of the coordinates of the manifold and standard deviation one.

As in §6.2, the proposed model was compared in terms of computational time and predictive performance with lasso and Cart. Table 3 and 4 show computational time and mean squared errors based on leave-one-out predictions considering each of the three simulation scenario. In particular, table 3 shows the results associated to the data drawn from the mixture of factor analyzers for different number of components. In this example, given the linear relationship between predictors and response, Lasso performs better than CART. Lasso is also much more efficient than CART and performs similarly to our model for moderately high dimensional features. However, as shown in table 3, as the sample size increases the computational time associated to Lasso dramatically increases compared to our model (see $p = 300,000$ for $n = 100, 200, 300$). Table 4 also shows that our model is associated to better predictive performance and CPU time.

7 Real application

We assessed the predictive performance of the proposed method on two very different neuroimaging datasets. First, we consider a structural connectome dataset collected at the Mind Research Network. Data were collected as described in Jung et al. [26]. We investigated the extent to which we could predict creative (as measured via the Composite Creativity Index [27]). For each subject, we estimate a 70 vertex undirected weighted brain-graph using the Magnetic Resonance Connectome Automated Pipeline [28] from diffusion tensor imaging data [29]. We therefore let each $x_i \in \Re^p$ correspond to logarithm of each weighted edge; because our graphs are undirected and lack self-loops, we have

Table 3: Non-linear manifold - MFA: Mean and standard deviations of squared errors under multi-scale stick-breaking (MSB), CART and Lasso for sample size 50 and 100 for different simulations sampled from a mixture of factor analyzers

				$N = 10$			$N = 5$	
p	n	SIM	MSB	CART	LASSO	MSB	CART	LASSO
$50e + 03$	100	MSE	0.23	0.42	0.36	0.17	0.43	0.22
		STD	0.34	0.59	0.43	0.18	0.69	0.23
		TIME	5	24	3	7	27	3
$50e + 03$	200	MSE	0.23	0.42	0.27	0.17	0.22	0.20
		STD	0.33	0.56	0.23	0.19	0.38	0.25
		TIME	10	51	8	12	56	7
$10e + 04$	100	MSE	0.67	1.35	1.32	0.15	0.17	0.22
		STD	1.04	2.26	1.36	0.23	0.19	0.23
		TIME	9	47	6	6	44	5
$10e + 04$	200	MSE	0.64	1.37	0.85	0.15	0.26	0.15
		STD	0.95	1.77	1.29	0.24	0.42	0.24
		TIME	15	99	15	11	89	15
$30e + 04$	100	MSE	0.26	0.39	0.31	0.63	1.40	1.01
		STD	0.39	0.51	0.52	0.80	1.24	1.46
		TIME	9.28	125	18	9	145	17
$30e + 04$	200	MSE	0.25	0.47	0.26	0.63	1.17	0.92
		STD	0.36	0.88	0.43	0.80	2.11	1.04
		TIME	15	262	40	13	283	43
$30e + 04$	300	MSE	0.25	0.30	0.30	0.62	1.42	0.70
		STD	0.36	0.41	0.48	0.89	1.85	0.94
		TIME	15	463	73	16	465	89

a total of $\binom{70}{2} = 2,415$ potential weighted edges. The vector of covariates consists in the natural logarithm of the total number of connections between all pairs of cortical regions, i.e. $p = 2,415$.

The second dataset comes from a resting-state functional magnetic resonance experiment as part of the Autism Brain Imaging Data Exchange [30]. We selected the Yale Child Study Center for analysis. Each brain-image was processed using the Configurable Pipepline for Analysis of Connectomes [31]. For each subject we computed a measure of normalized power at each voxel called fALFF [32]. fALFF is a highly nonlinear transformation of the time-series data, previously demonstrated to be a reliable property of such data. To ensure the existence of nonlinear signal relating these predictors, we let y_i correspond to an estimate of overall head motion in the scanner, called mean framewise displacement (FD) computed as described in Power et al. [33].

For the analysis, all variables were normalized by subtracting the mean and dividing by the standard deviation. The same prior specification and Gibbs sampler as in §5 was utilized. Table 5 shows mean and variance squared error based on leave-one-out predictions. Variable t_T is the amount of time necessary to obtain predictions for all subjects, while variables t_M and t_V are respectively the mean and the standard deviation of amount of time necessary to obtain one point predictions.

For the first data example, we compared our approach (multiresolution stick-breaking; MSB) to CART, lasso and random forests. Table 5 shows that MSB outperforms all the competitors in terms of mean square error; this is in addition to yielding an estimate of the entire conditional density for each y_i. It is also significantly faster that random forests, the next closest competitor, and faster than lasso. For this relatively low-dimensional example, CART is reasonably fast.

For the second data application, given the huge dimensionality of the predictor space, we were unable to get either CART or random forest to run to completion, yielding memory faults on our workstation (Intel Core i7-2600K Quad-Core Processor memory 8192 MB). We thus only compare

Table 4: Non-linear manifold - Swissroll and S-Manifold: Mean and standard deviations of squared errors under multiscale stick-breaking (MSB), CART and Lasso for sample size 50 and 100 for different simulation scenarios.

			SWISSROLL			S-MANIFOLD		
p	n		MSB	CART	LASSO	MSB	CART	LASSO
$10e+03$	100	MSE	0.25	0.46	0.38	0.67	0.70	0.77
		STD	0.24	0.53	0.40	0.76	0.80	0.85
		TIME	5	5	1	4	5	1
$10e+04$	50	MSE	0.24	0.44	0.25	0.38	0.38	0.84
		STD	0.24	0.42	0.29	0.40	0.35	0.80
		TIME	3	22	2	5	7	1
$10e+04$	100	MSE	0.24	0.43	0.17	0.25	0.30	0.70
		STD	0.26	0.55	0.22	0.22	0.25	0.50
		TIME	6	48	7	7	50	7
$20e+04$	50	MSE	0.24	0.67	0.29	0.35	0.40	0.73
		STD	0.23	0.50	0.29	0.22	0.30	0.40
		TIME	4	38	5	3	40	5
$20e+04$	100	MSE	0.25	0.78	0.33	0.37	0.37	0.70
		STD	0.26	0.74	0.36	0.25	0.27	0.55
		TIME	6	96	13	6	98	14
$50e+04$	50	MSE	0.17	0.47	0.23	0.16	0.20	0.35
		STD	0.23	0.43	0.22	0.20	0.19	0.40
		TIME	5	126	10	5	130	15
$50e+04$	100	MSE	0.17	0.33	0.19	0.11	0.25	0.56
		STD	0.21	0.46	0.23	0.14	0.20	0.61
		TIME	11	230	25	10	254	27

Table 5: Real Data: Mean and standard deviations of squared error under multiscale stick-breaking (MSB), CART, Lasso and random forest (RF).

DATA	n	p	MODEL	MSE	t_T	t_M	t_V
(1)	108	2,415	MSB	0.56	100	1.1	0.02
			CART	1.10	87	0.9	0.01
			LASSO	0.63	50	0.40	0.10
			RF	0.57	$7,817$	78.2	0.59
(2)	56	$10e+05$	MSB	0.76	690	20.98	2.31
			LASSO	1.02	$5,836$	96.18	9.66

performance to lasso. As in the previous example, MSB outperforms lasso in terms of predictive accuracy measured via mean-squared error, and significantly outperforms lasso in terms of computational time. Figure 4 shows the plot of CPU time used to predict each one of the 56 subjects involved in the experiment. The time needed to compute quantities utilized in all subject predictions was divided equally across subjects. Clearly, our approach is able to improve the computational time by up to five orders of magnitude.

8 Conclusion

We have proposed a new model which should lead to substantially improved predictive and computational performance to learn the density of a target variable given a high dimensional vector of predictors. As shown the proposed two stage approach can scale substantially better than other

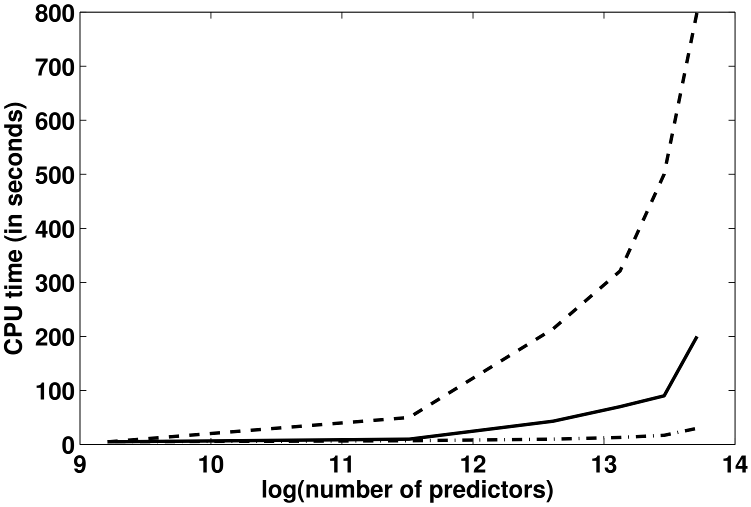

Figure 3: Elapsed CPU time (in seconds) for a single point prediction based on 200 observations for MSB (dot-dash), lasso (solid) and CART (dash) for different number of predictors in log-scale.

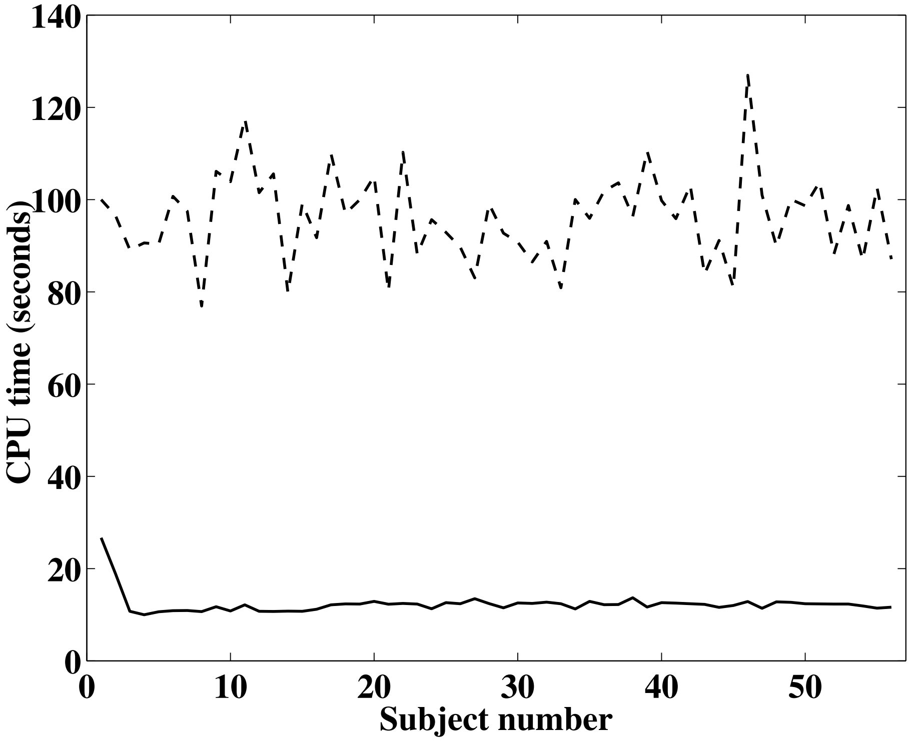

Figure 4: Plot of CPU time used to predict each one of the 56 measurement involved in experiment (2) under MSB (solid) and lasso (dash).

existing algorithms to massive number of features. We have focused on Bayesian MCMC-based methods, but there are numerous interesting directions for ongoing research. Moreover, in addition to better predictive and computational performance, our methods easily extend to parallelized and distributed systems, which we will also explore in future work.

References

[1] I. U. Rahman, I. Drori, V. C. Stodden, and D. L. Donoho. Multiscale representations for manifold- valued data. *SIAM J. Multiscale Model*, 4:1201–1232, 2005.

[2] W.K. Allard, G. Chen, and M. Maggioni. Multiscale geometric methods for data sets II: geometric wavelets. *Applied and Computational Harmonic Analysis*, 32:435–462, 2012.

[3] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixture of local experts. *Neural Computation*, 3:79–87, 1991.

[4] W. X. Jiang and M. A. Tanner. Hierarchical mixtures-of-experts for exponential family regression models: approximation and maximum likelihood estimation. *Annals of Statistics*, 27:987–1011, 1999.

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

[6] J. Q. Fan and T. H. Yim. A crossvalidation method for estimating conditional densities. *Biometrika*, 91:819–834, 2004.

[7] M. P. Holmes, G. A. Gray, and C. L. Isbell. Fast kernel conditional density estimation: a dual-tree Monte Carlo approach. *Computational statistics & data analysis*, 54:1707–1718, 2010.

[8] G. Fu, F. Y. Shih, and H. Wang. A kernel-based parametric method for conditional density estimation. *Pattern recognition*, 44:284–294, 2011.

[9] D. J. Nott, S. L. Tan, M. Villani, and R. Kohn. Regression density estimation with variational methods and stochastic approximation. *Journal of Computational and Graphical Statistics*, 21:797–820, 2012.

[10] M. N. Tran, D. J. Nott, and R. Kohn. Simultaneous variable selection and component selection for regression density estimation with mixtures of heteroscedastic experts. *Electronic Journal of Statistics*, 6:1170–1199, 2012.

[11] A. Norets and J. Pelenis. Bayesian modeling of joint and conditional distributions. *Journal of Econometrics*, 168:332–346, 2012.

[12] J. E. Griffin and M. F. J. Steel. Order-based dependent Dirichlet processes. *Journal of the American Statistical Association*, 101:179–194, 2006.

[13] D. B. Dunson, N. Pillai, and J. H. Park. Bayesian density regression. *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 69:163–183, 2007.

[14] D. B. Dunson, N.S. Pillai, and J. H. Park. Bayesian density regression. *Journal of the Royal Statistical Society*, 69:163–183, 2007.

[15] Y. Chung and D. B. Dunson. Nonparametric Bayes conditional distribution modeling with variable selection. *Journal of the American Statistical Association*, 104:1646–1660, 2009.

[16] S. T. Tokdar, Y. M. Zhu, and J. K. Ghosh. Bayesian density regression with logistic Gaussian process and subspace projection. *Bayesian Analysis*, 5:319–344, 2010.

[17] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. Chapman & Hall/CRC, 1984.

[18] J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19:1–141, 1991.

[19] L. Breiman. Bagging predictors. *Machine Learning*, 24:123140, 1996.

[20] R. Shapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26:16511686, 1998.

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

[22] I. Mossavat and O. Amft. Sparse bayesian hierarchical mixture of experts. *IEEE Statistical Signal Processing Workshop (SSP)*, 2011.

[23] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing 20*, 1:359392, 1999.

[24] J. Sethuraman. A constructive denition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[25] Didier Chauveau and Jean Diebolt. An automated stopping rule for mcmc convergence assessment. *Computational Statistics*, 14:419–442, 1998.

[26] Rex E Jung, Rachael Grazioplene, Arvind Caprihan, Robert S Chavez, and Richard J Haier. White matter integrity, creativity, and psychopathology: Disentangling constructs with diffusion tensor imaging. *PloS one*, 5(3):e9818, 2010.

[27] R. Arden, R. S. Chavez, R. Grazioplene, and R. E. Jung. Neuroimaging creativity: a psychometric view. *Behavioural brain research*, 214:143–156, 2010.

[28] William R. Gray, John A Bogovic, Joshua T. Vogelstein, Bennett A Landman, Jerry L Prince, and R. Jacob Vogelstein. Magnetic resonance connectome automated pipeline: an overview. *IEEE pulse*, 3(2):42–8, March 2010.

[29] Susumu Mori and Jiangyang Zhang. Principles of diffusion tensor imaging and its applications to basic neuroscience research. *Neuron*, 51(5):527–39, September 2006.

[30] Abide.

[31] Sharad Sikka, Joshua T. Vogelstein, and Michael Peter Milham. Towards Automated Analysis of Connectomes: The Configurable Pipeline for the Analysis of Connectomes (C-PAC). In *Organization of Human Brain Mapping*. Neuroinformatics, 2012.

[32] Qi-Hong Zou, Chao-Zhe Zhu, Yihong Yang, Xi-Nian Zuo, Xiang-Yu Long, Qing-Jiu Cao, Yu-Feng Wang, and Yu-Feng Zang. An improved approach to detection of amplitude of low-frequency fluctuation (ALFF) for resting-state fMRI: fractional ALFF. *Journal of neuroscience methods*, 172(1):137–141, July 2008.

[33] J. D. Power, K. A. Barnes, C. J. Stone, and R. A. Olshen. Spurious but systematic correlations in functional connectivity MRI networks arise from subject motion. *Neuroimage*, 59:2142–2154, 2012.