High Dimensional Nonparametric Regresssion via Additive Kernel Ridge Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

We describe additive kernel ridge regression (Add-KRR), a generalisation of the kernel ridge method for nonparametric regression. Nonparametric regression is exponentially difficult in high dimensions. It is difficult to make progress without making strong assumptions about the function. A common assumption in high dimensional regression models is to assume that the function is additive. In this work, we leverage this assumption, but considerably generalise existing additive models. We propose a convex optimisation objective for our problem. We compare several algorithms to optimise this objective on synthetic datasets. We also demonstrate that Add-KRRsignificantly outperforms other popular algorithms for nonparametric regression on moderate dimensional problems.

1 Introduction

Regression in high dimensions is an inherently difficult problem with known lower bounds depending exponentially in dimension [3]. In this project we intend to make progress in this problem by treating the function as an additive model of lower dimensional components. Using additive models is fairly standard in high dimensional regression literature [4, 5, 7]. However, in this work we wish to consider additive models which are more general/ expressive than previous work.

There are a number of potential nonparametric methods for modeling the low-order interaction terms. One option is to use multidimensional splines. Thin plate splines [11] extend spline-based nonparametric regression to multiple covariates, although computational complexity increases dramatically with the order of interaction. Natural thin plate splines extend one-dimensional smoothing splines. These can be fit via penalized regression, and complexity can be reduced by choosing knots on a grid rather than at each data point. Thin plate (penalized) regression splines can alternatively be used; this approach requires truncated SVD, but avoids choice of knots. Tensor product splines [9] can be used to construct multivariate splines as tensor products of single-dimensional splines. The number of basis functions grows exponential with the interaction order, but they can be fit via penalized regression.

Another option is to model the low-order interaction basis functions implicitly using kernels. There is existing research on using linear combinations of kernels for kernel learning, called multiple kernel learning [2]. Computationally efficient use of additive kernels has also been explored in Bayesian settings [1].

Our work extends Sparse Additive Models (SpAM) [7] to multi-dimensional nonparametric basis functions. For Sparse Additive Models, parameters are typically optimized using the backfitting algorithm. Our work also extends recent work on Generalized Additive Models plus Interactions [6]. However, in this work the interaction model was assumed to follow a specific functional form, leading to an optimization method tailored to their interaction model.

2 Problem Set up & Algorithm

2.1 Problem Statement & Notation

Let $f : \mathcal{X} \to \mathbb{R}$ be the function of interest. Here $\mathcal{X} \ni x = [x_1, \dots, x_D] \in \mathbb{R}^D$ and $\mathcal{X} \subset \mathbb{R}^D$. We have data $(X_i, Y_i)_1^n$ and wish to obtain an estimate $\hat{f}$ of f. In this work, we seek an additive approximation to the function. That is, $\hat{f}$ can be expressed as,

$$\hat{f}(x) = \hat{f}^{(1)}(x^{(1)}) + \hat{f}^{(2)}(x^{(2)}) + \cdots + \hat{f}^{(M)}(x^{(M)}), \tag{1}$$

where $x^{(j)} \in \mathcal{X}^{(j)} \subset \mathbb{R}^{d_j}$ and $\hat{f}^{(j)} : \mathcal{X}^{(j)} \to \mathbb{R}$. We shall refer to the $\mathcal{X}^{(j)}$'s as *groups* and the collection of all groups $\bigcup_{j=1}^{M} \mathcal{X}^{(j)}$ as the *decomposition*. We are particularly interested in the case where D is very large and the group dimensionality is bounded– i.e. $d_j \leq d \ll D$.

The work in Hastie and Tibshirani [4] treats $\hat{f}$ as a sum of one dimensional components. The decomposition here corresponds to $x^{(j)} = x_j$, $d_j = d = 1$ $\forall j$ and $M = D$. In this project, we would like to be more expressive than this model. We will consider decompositions for which $d > 1$ and more importantly allows for overlap between the groups. For e.g. $\hat{f}(x_1, x_2, x_3) = \hat{f}^{(1)}(x_1) + \hat{f}^{(2)}(x_1, x_2) + \hat{f}^{(3)}(x_2, x_3)$. Ravikumar et al. [7] treat $\hat{f}$ as a sparse combination of one dimensional functions. While this is seemingly restrictive than [4], the sparse approximation may provide favourable bias-variance tradeoffs in high dimensions. Drawing inspiration from this, we will consider models where M is very large and seek a sparse collection of groups to approximate the function - i.e. $\hat{f}^{(j)} = \mathbf{0}$ for several j.

2.2 Additive Kernel Ridge Regression

We begin with a brief review on Kernel Ridge Regression. One of several ways to formulate a nonparametric regression problem is to minimise an objective of the form $J(f) = \sum_{i=1}^{n} \ell(f(X_i), Y_i) + \lambda P(f)$ over a nonparametric class of functions $\mathcal{F}$. Here ℓ is a loss function and P is a term that penalises the complexity of the function f. Several nonparametric regression problems such as smoothing splines, natural splines and Kernel Ridge Regression can be written this way. Central to KRR is a positive semidefinite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ [8]. Then, $\mathcal{F}$ is taken to be the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ corresponding to k, P to be the squared RKHS norm of f and ℓ the squared error loss. Accordingly, KRR is characterised via,

$$\hat{f} = \operatorname*{argmin}_{f \in \mathcal{H}_k} \sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \|f\|_{\mathcal{H}_k}^2$$

However, as mentioned previously KRR suffers from the curse of dimensionality. To obtain an additive approximation using a given decomposition $\bigcup_j \mathcal{X}^{(j)}$, we consider kernels $k^{(j)} : \mathcal{X}^{(j)} \times \mathcal{X}^{(j)} \to \mathbb{R}$ acting on each group and their associated RKHSs $\mathcal{H}_{k^{(j)}}$. Further, since we will set M to be large and seek a sparse collection of functions in our additive model we introduce an additional penalty term for nonzero $f^{(j)}$. Putting it all together, our additive Kernel Ridge Regressin (Add-KRR) is characterised via the following problem where we jointly optimise over $f^{(1)}, \dots, f^{(M)}$,

$$\left(f^{(1)}, f^{(2)}, \dots, f^{(M)}\right) = \operatorname*{argmin}_{f^{(j)} \in \mathcal{H}_{k^{(j)}}, j=1,\dots,M} F\left(\{f^{(j)}\}_{j=1}^{M}\right) \quad \text{where,}$$

$$F\left(\{f^{(j)}\}_{j=1}^{M}\right) = \frac{1}{2} \sum_{i=1}^{n} \left(Y_i - \sum_{j=1}^{M} f^{(j)}(x^{(j)})\right)^2 + \frac{\lambda_1}{2} \sum_{j=1}^{M} \|f^{(j)}\|_{\mathcal{H}_{k^{(j)}}}^2 + \lambda_2 \sum_{j=1}^{M} \mathbb{1}(f^{(j)} \neq \mathbf{0}) \tag{2}$$

Our estimate for f is then $\hat{f}(\cdot) = \sum_j \hat{f}^{(j)}(\cdot)$.

Via an argument that uses the representer theorem it is straightforward to show that $f^{(j)}$ will be in the linear span of the reprodcing kernel maps of the training points ${X^{(j)}}_1^n$ – i.e. $f^{(j)}(\cdot) = \sum_j \alpha_i^{(j)} k^{(j)}(\cdot, X_i^{(j)})$. Then, the j^{th} term inside the summations of the second and third terms of equation (2) can be written as ${\alpha^{(j)}}^\top K^{(j)}\alpha^{(j)}$ and $\mathbb{1}(\alpha^{(j)} \neq 0)$. Here $K^{(j)} \in \mathbb{R}^{n\times n}\ \forall j$ such that $K^{(j)}_{rc} = k^{(j)}(X_r, X_c)$. Since the latter term is nonconvex we relax it via a group lasso type penalty. After simplifying the first term, our optimisation objective can be written as, $\boldsymbol{\alpha} = \operatorname{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^{nM}} F(\boldsymbol{\alpha})$ where,

$$F(\boldsymbol{\alpha}) = \frac{1}{2}\|Y - \sum_{j=1}^{m} K^{(j)}\alpha^{(j)}\|_2^2 + \frac{\lambda_1}{2}\sum_{j=1}^{M} {\alpha^{(j)}}^\top K^{(j)}\alpha^{(j)} + \lambda_2 \sum_{j=1}^{M} \|\alpha^{(j)}\|_2. \tag{3}$$

Here $\alpha^{(j)} \in \mathbb{R}^n\ \forall j$, $\boldsymbol{\alpha} = [{\alpha^{(1)}}^\top, \dots, {\alpha^{(M)}}^\top]^\top \in \mathbb{R}^{nM}$. Given the solution to the above, our estimate is obtained via $\hat{f}(\cdot) = \sum_{j=1}^{M}\sum_{i=1}^{n} \alpha_i^{(j)} k^{(j)}(\cdot, X_i^{(j)})$. Equation (3) will the (convex) optimisation problem in our algorithm.

2.3 Practical Considerations

All that is left to do to complete the specification of our algorithm is to describe the allocation of coordinates into different groups and the kernel used for each such group. For this we tried 3 different strategies, all of which require the specification of a group size parameter d. The first sets $M = \binom{D}{d}$ and uses all combinations of size d groups. The second sets $M = \sum_{k=1}^{d}\binom{D}{k}$ and uses all combinations of up to size d. In the third we also specify M and then randomly generate M groups of size d. All three performed equally well so we only consider the third option as M does not grow combinatorially with D and d.

As is the case in several kernel methods, the choice of the kernel is important for good empirical performance. For each $k^{(j)}$ we use an RBF kernel, with scale parameter σ_j and bandwidth parameter h_j.

$$k^{(j)}(x_s^{(j)}, x_t^{(j)}) = \sigma_j \exp\left(\frac{\|x_s^{(j)} - x_t^{(j)}\|_2^2}{2h_j^2}\right)$$

Here σ_j captures the variation in the output and h_j captures the variation in the input. In KRR, these parameters along with the penalty coefficient λ are chosen via cross validation. However, this is infeasible in our setting as M is potentially very large. In our case we set $h_j = 1.5\|\mathrm{std}({X^{(j)}}_1^n)\|_2 n^{\frac{-1}{4+d_j}}$. Here $\mathrm{std}({X^{(j)}}_1^n)$ is the vector of standard deviations of the training dataset on the coordinates belonging to group j. This choice of bandwidth is motivated by the Silverman bandwidth and several kernel methods which choose a bandidth on the order $O(n^{\frac{-1}{2\beta+p}})$ where β is the smoothness of the function [10] and p is the dimensionality. We set $\sigma_j = \mathrm{std}(Y)/\sqrt{M}$ to capture the per group variance. The performance of the algorithm was usually insensitive to the choices of h_j and σ_j provided they were roughly in the correct range. The penalty coefficients λ_1 and λ_2 are chosen via cross validation.

3 Optimisation Methods

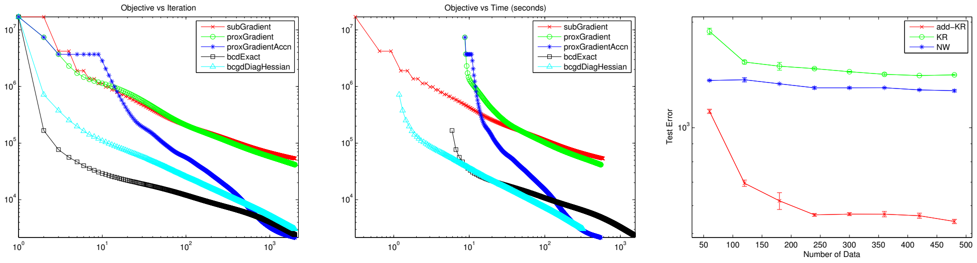
SubGradient Method

Proximal Gradient Method

Note that we can write the objective (3) as $F(\boldsymbol{\alpha}) = G(\boldsymbol{\alpha}) + \Psi(\boldsymbol{\alpha})$ where G is smooth and Ψ is not. Ψ is the popular group lasso penalty. Via the Moreau decomposition, and using the fact that the argument in the prox operator is separable, the prox operator can be shown to be,

$$[\mathrm{prox}_{\Psi,t}(\boldsymbol{\alpha})]^{(j)} = \mathrm{prox}_{\Psi,t}(\alpha^{(j)}) = \begin{cases} \alpha^{(j)} - t\frac{\alpha}{\|\alpha^{(j)}\|_2} & \text{if } \|\alpha^{(j)}\| < t \\ \mathbf{0} & \text{otherwise} \end{cases}$$

Obtaining the prox operator takes $O(nM)$ time and has cost comparable to computing the gradient of $G(\boldsymbol{\alpha})$. We use the above to implement proximal gradient method as described in the class notes. We use backtracking to determien the step size and experiment both with and without acceleration.



(a) (b) (c)

Figure 1: Figures (a) and (b) plot the objective value vs iteration and cpu time respectively. Accelerated proximal gradient descent (blue curve) seems to perform best. Figure (c) plots the test set error for Add-KRR , KRR and NW on a synthetic problem. Add-KRR outperforms both methods. We apologise for the small labels as we were running short of time.

BCD - use full name

BCGD

4 Experiments

In this section, we wish to experimentally evaluate two criteria: the performance of the different optimisation algorithms to minimise the Add-KRR objective (3), and the performance of Add-KRR and other nonparametric regression methods. So far, we only have results on synthetic data.

Experimental set up

For our synthetic experiments, we take $\mathcal{X} = [-1, 1]^D$. We constructed a function of 3 modes by taking the logarithm of 3 Gaussian bumps whose centres are randomly chosen from $\mathcal{X}$. We uniformly randomly sample 500 points each for training and testing from $\mathcal{X}$ and uses the true function values as the labels. In the examples we consider below we take $D = 20$, $d = 4$ and $M = 50$. Therefore, our optimisation problem had $50 \times 500 = 25,000$ parameters.

Comparison of Optimisation Algorithms

We compared the five methods specified above on the synthetic example outlined above. Figures 1(a) and (b) shows the improvement in the objective over iterations and time respectively over 2000 iterations. subGradient and proxGradient exhibit slow convergence but proxGradientAccn , bcdExact and bcdDiagHessian perform quite well. proxGradientAccn does marginally better than the rest. Even for different values of D, d, M we found that the latter three methods did better but their relative performances varied slightly. In our experiments below we use proxGradientAccn .

Comparison with other Regression Algorithms

We use the same experimental set up as explained above to compare Add-KRR against the KRR and NW methods. To choose the kernel bandwidth for NW we used 5-fold cross validation. Due to time constraints, parameter selection for Add-KRR and KRR was very coarse – we tried only 5 different choices for (λ_1, λ_2) and used 2-fold cross validation. Despite this, Add-KRR significantly ourperforms both KRR and NW on the synthetic problem. The results are depicted in Figure 1(c). These results are quite encouraging as they attest well to our intuitions about favourable bias variance tradeoffs when using (statistically simpler) additive models for high dimensional regression.

5 Conclusion

Our initial results using the proposed method is fairly promising. Going forward we wish to perform comparisons with other nonparametric regression methods such as Gaussian process regression, locally polynomial regression and other additive models [1, 4, 7] on other synthetic and real datasets. One challenge will be in scaling this to large datasets as we need to construct several $n \times n$ kernel matrices. We wish to explore some ideas on this front as well.

References

[1] David K. Duvenaud, Hannes Nickisch, and Carl Edward Rasmussen. Additive gaussian processes. In *Advances in Neural Information Processing Systems*, 2011.

[2] Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

[3] László Györfi, Micael Kohler, Adam Krzyzak, and Harro Walk. *A Distribution Free Theory of Nonparametric Regression*. Springer Series in Statistics, 2002.

[4] T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*. London: Chapman & Hall, 1990.

[5] John D. Lafferty and Larry A. Wasserman. Rodeo: Sparse Nonparametric Regression in High Dimensions. In *NIPS*, 2005.

[6] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631. ACM, 2013.

[7] Pradeep Ravikumar, John Lafferty, Han Liu, and Larry Wasserman. Sparse Additive Models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2009.

[8] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[9] Charles J Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *The Annals of Statistics*, pages 118–171, 1994.

[10] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Publishing Company, Incorporated, 2008.

[11] Simon N Wood. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):95–114, 2003.