Penalized Additive Least Squares Models for High Dimensional Nonparametric Regression and Function Selection

Abstract

We describe additive kernel regression (Add-KR), a generalisation of kernel least squares methods 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 and optimise it using Block Coordinate Gradient Descent. We demonstrate that Add-KR significantly outperforms other popular algorithms for nonparametric regression on moderate to high dimensional problems.

1. Introduction

Nonparametric regression in high dimensions is an inherently difficult problem with known lower bounds depending exponentially in dimension (Györfi et al., 2002). With rare exceptions, nonparametric methods typically work well only under at most $4-6$ dimensions. In this project we intend to make progress in this problem by treating the *estimate* of the function as an additive function of lower dimensional components.

Using additive models is fairly standard in high dimensional regression literature (Hastie & Tibshirani, 1990; Ravikumar et al., 2009; Lafferty & Wasserman, 2005). When the true underlying function f exhibits additive structure, using an additive model for estimation is understandably reasonable. However, even when f is not additive, using an additive model has its advantages. It is a well understoon notion in Statistics that when we only have a few samples, using a simpler model to fit our data may give us a better tradeoff for estimation error against approximation error. This is because additive functions are *statistically simpler* than more general (non-additive) functions. Typically, in most nonparametric regression methods using kernels such as the Nadaraya-Watson (NW) estimator and Kernel Ridge Regression (KRR), the bias-variance tradeoff is managed via the bandwidth of the kernel. Using an additive model provides another "knob" to control this tradeoff and provides significant gains in high dimensional regression. In fact, Duvenaud et al. (2011) demonstrate that using additive models in the Gaussian Process (GP) framework improves prediction performance.

Our methods are based on KRR, where we choose to model the low-order interaction implicitly using kernels acting on subsets of the coordinates (groups). We minimize the squared-error loss with a squared RKHS norm penalty to enforce smoothness and a mixed $\ell_{1,2}$-norm (group lasso style) penalty to enforce a sparse collection of functions. This leads to a convex objective function where the number of parameters is the product of the number of samples and the number of basis functions.

Our work extends Sparse Additive Models (SpAM) (Ravikumar et al., 2009) to multidimensional nonparametric basis functions. Our proposed method also extends recent work on Generalized Additive Models plus Interactions (Lou et al., 2013). 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. Our research is also related to existing work on using linear combinations of kernels for kernel learning, called multiple kernel learning (Gönen & Alpaydın, 2011).

Optimization for our proposed method is complicated by the non-smooth $\ell_{1,2}$-norm regularization penalty. Algorithms for group lasso have addressed this problem through a variety of approaches. Proximal gradient (Beck & Teboulle, 2009) has cheap iterations and relatively fast convergence if combined with acceleration. An block coordinate descent method has also been developed (Qin et al., 2013). Further, the general Coordinate Gradient Descent method (Tseng & Yun, 2009) can also be specialized to $\ell_{1,2}$-penalized problems (Meier et al., 2008; Friedman et al., 2010). Recent work (Wytock et al., 2014) on the group fused lasso has sidestepped the $\ell_{1,2}$-norm penalty, transforming it to a smooth objective with non-negativity constraint. Finally, safe screening rules for the group lasso have been developed (Wang et al., 2013) which quickly eliminate many of the all-0 groups. For Sparse Additive Models, parameters are typically optimized via the back-

fitting algorithm (Ravikumar et al., 2009), a special case of (block) coordinate descent with group sizes of 1. In our work, we experiment with several optimisation methods for non-smooth objectives. In our experiments, Block Coordinate Gradient Descent provided the best performance.

2. Problem Set up & Algorithm

2.1. Problem Statement & Notation

Let $f : \mathcal{X} \to \mathbb{R}$ be the function of interest. Here $\mathcal{X} \ni x = [x_1, \ldots, x_D] \in \mathbb{R}^D$ 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,

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

where each $\hat{f}^{(j)} : \mathcal{X} \to \mathbb{R}$.

The work in Hastie & Tibshirani (1990) treats $\hat{f}$ as a sum of one dimensional components. In Equation (1) this corresponds to setting $M = D$ and have each $\hat{f}^{(j)}$ act on only the j^{th} coordinate. In this work, we would like to be more expressive than this model. We will consider additive models on more than just one dimension 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. (2009) treat $\hat{f}$ as a sparse combination of one dimensional functions. While this is seemingly restrictive than (Hastie & Tibshirani, 1990), the sparse approximation may provide favourable bias-variance tradeoffs in high dimensions. Drawing inspiration from this, we will also 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 Least Squares Regression via Kernels

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 Gaussian processes, smoothing splines and natural splines can be formulated this way. Or particular interest to us is Kernel Ridge Regression (KRR) which uses a positive semidefinite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (Scholkopf & Smola, 2001) and takes $\mathcal{F}$ is taken to be the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ corresponding to k. P is taken to be the squared RKHS norm of f and ℓ the squared error loss. Accordingly, KRR is characterised via the optimisation objective,

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

However, like most nonparametric regression models, KRR suffers from the curse of dimensionality. To obtain an additive approximation we consider M kernels $k^{(j)}$ and their associated RKHSs $\mathcal{H}_{k^{(j)}}$. In equation (1), we will aim for $\hat{f}^{(j)} \in \mathcal{H}_{k^{(j)}}$. Accordingly we consider an optimisation problem of the following form where we jointly optimise over $\hat{f}^{(1)}, \ldots, \hat{f}^{(M)}$,

$$\{\hat{f}^{(j)}\}_{j=1}^M \operatorname*{argmin}_{f^{(j)} \in \mathcal{H}_{k^{(j)}}, j=1,\ldots,M} F\left(\{f^{(j)}\}_{j=1}^M\right)$$

where, (2)

$$F\left(\{f^{(j)}\}_{j=1}^M\right) = \frac{1}{2}\sum_{i=1}^n \left(Y_i - \sum_{j=1}^M f^{(j)}(x^{(j)})\right)^2 + \lambda \sum_{j=1}^M \|f^{(j)}\|_{\mathcal{H}_{k^{(j)}}}^q$$

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

Via a representer theorem like argument it is straightforward to show that $f^{(j)}$ will be in the linear span of the reprodcing kernel maps of the training points X_1^n – i.e. $f^{(j)}(\cdot) = \sum_j \alpha^{(j)}{}_i k^{(j)}(\cdot, X_i)$. Then, the j^{th} term in the second summation can be written as ${\alpha^{(j)}}^\top K^{(j)} \alpha^{(j)}$, where $K^{(j)} \in \mathbb{R}^{n \times n}$ $\forall j$ such that ${K^{(j)}}_{rc} = k^{(j)}(X_r, X_c)$. After further simplification, the objective can be written as, $\boldsymbol{\alpha} = \operatorname{argmin}_{\boldsymbol{\alpha} \in \mathbb{R}^{nM}} F(\boldsymbol{\alpha})$ where,

$$F_1(\boldsymbol{\alpha}) = \frac{1}{2}\left\|Y - \sum_{j=1}^m K^{(j)}\alpha^{(j)}\right\|_2^2 + \lambda \sum_{j=1}^M \left({\alpha^{(j)}}^\top K^{(j)} \alpha^{(j)}\right)^{q/2}. \qquad (3)$$

Here $\alpha^{(j)} \in \mathbb{R}^n$ $\forall j$, $\boldsymbol{\alpha} = [{\alpha^{(1)}}^\top, \ldots, {\alpha^{(M)}}^\top]^\top \in \mathbb{R}^{nM}$ and $Y = [Y_1, \ldots, Y_n]^\top \in \mathbb{R}^n$. Given the solution to the above, our estimate is obtained via $\hat{f}(\cdot) = \sum_{j=1}^M \sum_{i=1}^n \alpha^{(j)}{}_i k^{(j)}(\cdot, X_i^{(j)})$. Equation (3) will the (convex) optimisation problem in our algorithm. We call this algorithm Additive Kernel Regression (Add-KR). A natural choice for q in the objective (2) is $q = 2$. However in this work we use $q = 1$ since it encourages a sparse subset of functions as the solution which provides interpretability of the learned models.

2.3. Choice of Kernels

All that is left to do to complete the specification of our algorithm is to describe the construction of the kernels $k^{(j)}$. We consider two settings in this regard.

The first is when we wish to reduce the statistical complexity of the function we to be learned in high dimen-

sion. A kernel directly defined on D dimensions is complex since it allows for interactions of all D variables. We may reduce the complexity of the kernel by constraining how these variables interact. In particular we consider kernels of the form,

$$k^{(1)}(x,x') = \sum_{1\le i\le D} k_i(x_i,x'_i) \tag{4}$$
$$k^{(2)}(x,x') = \sum_{1\le i_1<i_2\le D} k_{i_1}(x_{i_1},x'_{i_1})k_{i_2}(x_{i_2},x'_{i_2})$$
$$k^{(M)}(x,x') = \sum_{1\le i_1<i_2<\cdots<i_M\le D}\ \prod_{d=1}^{M} k_{i_d}(x_{i_d},x'_{i_d})$$

Here $k_i : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$ is a base kernel acting on one dimension. $k^{(j)}$ has $\binom{D}{j}$ terms and exhaustively computing all of them is computationally intractable. Fortunately, by observing that the j^{th} kernel is just the j^{th} elementary symmetric polynomial (ESP) on the base kernel values we may use the Newton Girard formula to efficiently compute them recursively. Precisely, by denoting $\kappa_s = \sum_{i=1}^{D}(k_i(x_i,x'_i))^s$ we have,

$$k^{(j)}(x,x') = \frac{1}{j}\sum_{d=1}^{j}(-1)^{d-1}\ \kappa_j\ k^{(j-d)}(x,x')$$

Computing the M kernels this way only requires $O(DM)$ computation. We call this choice of kernels the *ESP Kernels*. A similar kernel using a similar trick for computing it was used in Duvenaud et al. (2011).

The second setting is when we are explicitly searching for a sparse subset of functions to explain the data. For instance, in neurological models, while the function of interest has several variables the interactions are sparse and of lower order. For example, a function of 4 variables may take the form

$$f(x) = f^{(1)}(x_1) + f^{(2)}(x_2,x_3) + f^{(3)}(x_1,x_4)$$

That is, the function decomposes as a sum of functions acting on small groups of variables. Given a large set of candidate groups, the task at hand is to recover the groups and the individual functions acting on those groups. In this setting, M and our RKHSs are determined by the problem – $\mathcal{H}_{k^{(j)}}$ contains functions on the variables belonging to the j^{th} candidate group.

2.4. Implementation

We now describe the implementation and other detials of the above algorithm. Let the Cholesky decomposition of $K^{(j)}$ be $K^{(j)} = L^{(j)}{L^{(j)}}^\top$. Denote $\beta^{(j)} = {L^{(j)}}^\top\alpha^{(j)}$. Then, our objective can be written in terms of $\boldsymbol{\beta} = [{\beta^{(1)}}^\top,\ldots,{\beta^{(M)}}^\top]$ as,

$$F_2(\boldsymbol{\beta}) = \frac{1}{2}\Big\|Y - \sum_{j=1}^{m} L^{(j)}\alpha^{(j)}\Big\|_2^2 + \lambda\sum_{j=1}^{M}\|\beta^{(j)}\|_2 \tag{5}$$

The objective, in the above form is well studied in optimisation literature as the group LASSO. When the number of parameters for each group are small, which is typically the case in group LASSO problems, block coordinate descent (BCD) is believed to be the state of the art solver. However, in our case the number of parameters is large – growing linearly with n. In this regime BCD is slow since it requires a matrix inversion at each step. In particular, we found that Block Coordinate Gradient Descent (BCGD) significantly outperformed BCD in our experiments. In fact, we tried several other methods including subgradient method, proximal gradient method and ADMM and found that BCGD performed best.

The penalty term λ was chosen using 5-fold cross validation. Our implementation first solves for the largest λ value. For successive λ values, we initialise BCGD at the solution of the previous λ value. This warm starts procedure significantly speeds up the running time of the entire training procedure.

3. Experiments

We present empirical results on synthetic and real datasets in both settings described above.

3.1. Setting 1: ESP Kernels

In our implementations, for the one dimensional base kernel we use the RBF kernel $k_i(x,x') = \exp((x-x')^2/h^2)$ with bandwidth h. Since cross validating on all the kernel bandwidths is expensive, we set it to $h = c\sigma n^{-0.2}$. This follows other literature (Györfi et al., 2002; Ravikumar et al., 2009) using similar choices for kernel bandwidths. The constant c was hand tuned – we found that the performance of our methods was robust to choices of c between 5 and 40. The value of M was also hand tuned and set to $M = \min(D/4, 10)$.

We compare Add-KR against kernel kidge regression(KRR), Nadaraya Watson regression (NW), locally linear regression (LL), locally quadratic regression (LQ), Gaussian process regression (GP), k nearest neighbors regression (kNN) and support vector regression (SVR). For GP and SVR we use the implementations in Rasmussen & Nickisch (2010); Chang & Lin (2011) respectively. For the other methods, we chose hyper parameters using 5-fold cross validation. The Additive Gaussian process model of Duvenaud et al. (2011) is also a candidate but we found that inference was extremely slow beyond a few hundred train-

Dataset (D, n)	Add-KR	KRR	kNN	NW	LL	LQ	GP	SVR
Speech (21, 520)	**0.02269**	0.02777	0.09348	0.11207	0.03373	0.02407	0.02531	0.22431
Music (90, 1000)	**0.91627**	0.91922	1.00001	1.05745	1.25805	1.06482	0.94329	1.07009
Tele-motor (19, 300)	**0.06059**	0.06488	0.13957	0.20119	0.09455	0.08774	0.06678	0.38038
Housing (12, 256)	**0.31285**	0.35947	0.43619	0.42087	**0.31219**	0.35061	0.67566	1.15272
Blog (91, 700)	**1.43288**	1.53227	1.73545	1.49305	1.69234	1.71321	1.64429	1.66705
Forest Fires (10, 210)	0.30675	0.32618	0.40565	0.37199	0.35462	0.33881	**0.29038**	0.70154
Propulsion (15, 400)	0.04167	0.01396	0.15760	0.11237	0.182345	0.19212	**0.00355**	0.74511

Table 1. The test set errors of all methods on 7 datasets from the UCI repository. The dimensionality and number of training points is indicated next to the dataset. The best method(s) for each dataset are in bold font. Add-KRgives the best results in most of the datasets and is within the top 3 in all of the datasets. In the Forest Fires dataset it is only slightly worse than GP. In the Propulsion dataset, GPseems to significantly outperform all other methods.

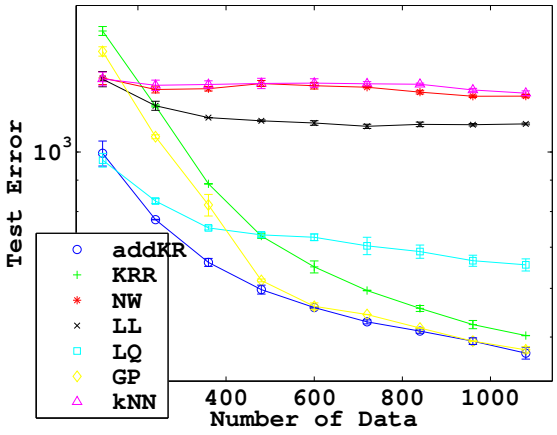

Figure 1. Comparison of various nonparametric regression methods on a 20-dimensional toy dataset. The x-axis denotes the number of training data and the y-axis is the test error.

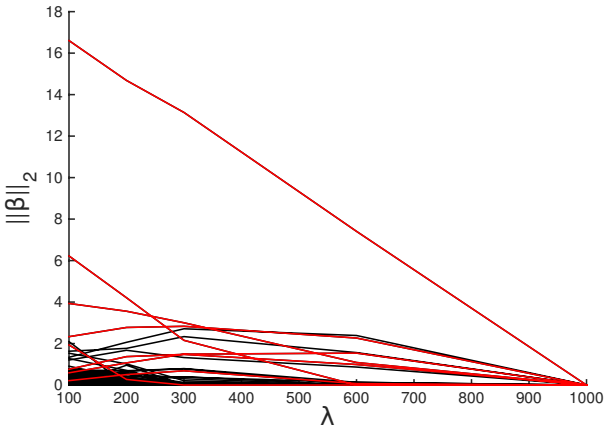

Figure 2. Solution path with $n = 200$ samples (above) and $n = 600$ (below). The x-axis shows the regularization parameter, while the y-axis plots $\|f^{(j)}\|_{\mathcal{H}_{k^{(j)}}} = \|\beta^{(j)}\|$. The true nonzero functiosn are depicted in red. As the figure indicates, several of the false functions are driven to 0 fast whereas the true functions persist for longer.

ing points (For e.g. it took > 50 minutes with 600 points whereas Add-KR ran in under 4 minutes).

First, we construct a smooth synthetic 20 dimensional function. We train all methods on n training points where n varies from 100 to 1100 and test on 1000 points sampled independently. The results are shown in Figure 1. Add-KR outperforms all other methods. We suspect that NW, LL and kNN perform very poorly since they make very weak smoothness assumptions about the function.

Next, we compare all methods on 7 moderate to high dimensional datasets from the UCI repository. All inputs and labels were preprocessed to have zero mean and unit standard deviation. We split the datasets into roughly two halves for training and testing. The results are given in Table 1. Our proposed method outperforms alternatives in most cases.

3.2. Setting 2: Function Selection

In this section, we study the ability of our method to recover the true function. We use RBF kernels on each group by setting kernel bandwidths for each dimension as same as explained above. We generate 600 observations from the following 50-dimensional additive model:

$$\begin{aligned} y_i =& f_1(x_{i1}) + f_2(x_{i2}) + f_3(x_{i3}) + f_4(x_{i4}) + \\ & f_1(x_{i5}x_{i6}) + f_2(x_{i7}x_{i8}) + \\ & f_3(x_{i9}x_{i10}) + f_4(x_{i11}x_{i12}) + \epsilon_i \end{aligned}$$

where,

$$\begin{aligned} f_1(x) =& -2\sin(2x), f_2(x) = x^2 - \frac{1}{3}, \\ f_3(x) =& x - \frac{1}{2}, f_4(x) = e^{-x} + e^{-1} - 1 \end{aligned}$$

with noise $\epsilon_i \sim \mathcal{N}(0, 1)$. Thus, 46 out of 50 individual features are irrelevant, and 1221 out of 1225 pairwise features

are irrelevant. As candidates, we use all functions of first and second order interactions – i.e the kernels characterising our RKHSs are of the form $k(x_i, x_i')$ for $i = 1, \ldots, 50$ and $k(x_i, x_i)k(x_j, x_j)$ for $1 \leq i < j \leq 50$. Therefore, in this experiment $M = 1275$.

We plot the solution path for two independent datasets. The plots give the RKHS norm of the function on each kernel $\|\hat{f}^{(j)}\|_{\mathcal{H}_{k^{(j)}}} = \|\beta^{(j)}\|_2$ for all kernels against the value of the regularization parameter λ. The results are shown in Figure 2. At $\lambda = 200$ we recover all true nonzero functions for a true positive rate of 100% and have 47 false negatives for a false positive rate of 3.7%

4. Conclusion

We proposed a framework for additive least squares regression. We design our estimate to be a sum of functions where the functions are obtained by jointly optimising over several RKHSs. The proposed framework is useful for high dimensional nonparametric regression since it provides favourable bias variance tradeoffs in high dimensions. Further, it can also be used for the recovery of sparse functions when the underlying function is additive. Our initial experimental results indicate that our methods are superior or competitive with existing methods in both fronts.

Going forward, we wish to study the theoretical properties of such penalized additive models especially focusing on rate of convergence and sparsistency.

References

Beck, Amir and Teboulle, Marc. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Chang, Chih-Chung and Lin, Chih-Jen. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

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

Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

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

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

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

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

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

Meier, Lukas, Van De Geer, Sara, and Bühlmann, Peter. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

Qin, Zhiwei, Scheinberg, Katya, and Goldfarb, Donald. Efficient block-coordinate descent algorithms for the group lasso. *Mathematical Programming Computation*, 5(2): 143–169, 2013.

Rasmussen, Carl Edward and Nickisch, Hannes. Gaussian Processes for Machine Learning (GPML) Toolbox. *J. Mach. Learn. Res.*, 2010.

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

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

Tseng, Paul and Yun, Sangwoon. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1-2):387–423, 2009.

Wang, Jie, Zhou, Jiayu, Wonka, Peter, and Ye, Jieping. Lasso screening rules via dual polytope projection. In *Advances in Neural Information Processing Systems*, pp. 1070–1078, 2013.

Wytock, Matt, Sra, Suvrit, and Kolter, J Zico. Fast newton methods for the group fused lasso. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, 2014.