Fast Algorithms for Segmented Regression

Fast Algorithms for Segmented Regression

Jayadev Acharya JAYADEV@MIT.EDU
Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Ilias Diakonikolas DIAKONIK@USC.EDU
University of Southern California, Los Angeles, MA 90089, USA

Jerry Li JERRYZLI@MIT.EDU
Ludwig Schmidt* LUDWIGS@MIT.EDU
Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Abstract

We study the fixed design segmented regression problem: Given noisy samples from a piecewise linear function f, we want to recover f up to a desired accuracy in mean-squared error.

Previous rigorous approaches for this problem rely on dynamic programming (DP) and, while sample efficient, have running time quadratic in the sample size. As our main contribution, we provide new sample near-linear time algorithms for the problem that – while not being minimax optimal – achieve a significantly better sample-time tradeoff on large datasets compared to the DP approach. Our experimental evaluation shows that, compared with the DP approach, our algorithms provide a convergence rate that is only off by a factor of 2 to 4, while achieving speedups of three orders of magnitude.

1. Introduction

We study the *regression* problem – a fundamental inference task with numerous applications that has received tremendous attention in machine learning and statistics during the past fifty years (see, e.g., (Mosteller & Tukey, 1977) for a classical textbook). Roughly speaking, in a (fixed design) regression problem, we are given a set of n observations $(\mathbf{x}_i, y_i)$, where the y_i's are the dependent variables and the $\mathbf{x}_i$'s are the independent variables, and our goal is to model the relationship between them. The typical assumptions are that (i) there exists a simple function f that (approximately) models the underlying relation, and (ii) the dependent observations are corrupted by random noise. More specifically, we assume that there exists a family of functions $\mathcal{F}$ such that for some $f \in \mathcal{F}$ the following holds: $y_i = f(\mathbf{x}_i) + \epsilon_i$, where the ϵ_i's are i.i.d. random variables drawn from a "tame" distribution such as a Gaussian (later, we also consider model misspecification).

Throughout this paper, we consider the classical notion of Mean Squared Error (MSE) to measure the performance (risk) of an estimator. As expected, the minimax risk depends on the family $\mathcal{F}$ that f comes from. The natural case that f is linear is fully understood: It is well-known that the least-squares estimator is statistically efficient and runs in sample-linear time. The more general case that f is *non-linear*, but satisfies some well-defined structural constraint has been extensively studied in a variety of contexts (see, e.g., (Gallant & A., 1973; Feder, 1975; Friedman, 1991; Bai & Perron, 1998; Yamamoto & Perron, 2013; Kyng et al., 2015; Avron et al., 2013; Meyer, 2008; Chatterjee et al., 2015)). In contrast to the linear case, this more general setting is not well-understood from an information-theoretic and/or computational aspect.

In this paper, we focus on the case that the function f is promised to be *piecewise linear* with a given number k of *unknown* pieces (segments). This is known as fixed design *segmented* regression, and has received considerable attention in the statistics community (Gallant & A., 1973; Feder, 1975; Bai & Perron, 1998; Yamamoto & Perron, 2013). The special case of piecewise polynomial functions (splines) has been extensively used in the context of inference, including density estimation and regression, see, e.g., (Wegman & Wright, 1983; Friedman, 1991; Stone, 1994; Stone et al., 1997; Meyer, 2008).

Information-theoretic aspects of the segmented regression problem are well-understood: Roughly speaking, the min-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

*Authors ordered alphabetically.

imax risk is inversely proportional to the number of samples. In contrast, the computational complexity of the problem is poorly understood: Prior to our work, known algorithms for this problem with provable guarantees were quite slow. Our main contribution is a set of new *provably fast* algorithms that outperform previous approaches both in theory and in practice. Our algorithms run in time that is nearly-linear in the number of data points n and the number of intervals k. Their computational efficiency is established both theoretically and experimentally. We also emphasize that our algorithms are robust to model misspecification, i.e., they perform well even if the function f is only *approximately* piecewise linear.

Note that if the segments of f were known a priori, the segmented regression problem could be immediately reduced to k independent linear regression problems. Roughly speaking, in the general case (where the location of the segment boundaries is unknown) one needs to "discover" the right segments using information provided by the samples. To address this algorithmic problem, previous works (Bai & Perron, 1998; Yamamoto & Perron, 2013) relied on dynamic programming that, while being statistically efficient, is computationally quite slow: its running time scales at least *quadratically* with the size n of the data, hence it is rather impractical for large datasets.

Our main motivation comes from the availability of large datasets that has made computational efficiency the main bottleneck in many cases. In the words of (Jordan, 2013): "As data grows, it may be beneficial to consider faster inferential algorithms, because the increasing statistical strength of the data can compensate for the poor algorithmic quality." Hence, it is sometimes advantageous to sacrifice statistical efficiency in order to achieve faster running times because we can then achieve the desired error guarantee faster (provided more samples). In our context, instead of using a slow dynamic program, we employ a subtle iterative greedy approach that runs in sample-linear time.

Our iterative greedy approach builds on the work of (Acharya et al., 2015; **?**), but the details of our algorithms here and their analysis are substantially different. In particular, as we explain in the body of the paper, the natural adaptation of their analysis to our setting fails to provide any meaningful statistical guarantees. To obtain our results, we introduce novel algorithmic ideas and carefully combine them with additional probabilistic arguments.

2. Preliminaries

In this paper, we study the problem of fixed design segmented regression. We are given samples $\boldsymbol{x}_i \in \mathbb{R}^d$ for $i \in [n] (= \{1, \ldots, n\})$, and we consider the following classical regression model:

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i \ . \tag{1}$$

Here, the ϵ_i are i.i.d. sub-Gaussian noise variables with variance proxy σ^2, mean $\mathbb{E}[\epsilon_i] = 0$, and variance $s^2 = \mathbb{E}[\epsilon_i^2]$ for all i. We will let $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ denote the vector of noise variables. We also assume that $f : \mathbb{R}^d \to \mathbb{R}$ is a k-piecewise linear function. Formally, this means:

Definition 1. *The function $f : \mathbb{R}^d \to \mathbb{R}$ is a* k-piecewise linear *function if there exists a partition of the real line into k disjoint intervals $I_1, \ldots, I_k$, k corresponding parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k \in \mathbb{R}^d$, and a fixed, known j such that for all $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ we have that $f(\boldsymbol{x}) = \langle \boldsymbol{\theta}_i, \boldsymbol{x} \rangle$ if $x_j \in I_i$. Let $\mathcal{L}_{k,j}$ denote the space of k-piecewise linear functions with partition defined on coordinate j.*

Moreover, we say f is flat *on an interval $I \subseteq \mathbb{R}$ if $I \subseteq I_i$ for some $i = 1, \ldots, k$, otherwise, we say that f has a* jump *on the interval I.*

In the full paper, we also discuss the agnostic setting where the ground truth f is not piecewise linear itself but only well-approximated by a k-piecewise linear function. For simplicity of exposition, we assume that the partition coordinate j is 1 in the rest of the paper. We remark that this model also contains the problem of (fixed design) piecewise polynomial regression as an important subcase (see the full paper for details).

Following this generative model, a regression algorithm receives the n pairs $(\boldsymbol{x}_i, y_i)$ as input. The goal of the algorithm is then to produce an estimate $\widehat{f}$ that is close to the true, unknown f with high probability over the noise terms ϵ_i and any randomness in the algorithm. We measure the distance between our estimate $\widehat{f}$ and the unknown function f with the classical mean-squared error:

$$\mathrm{MSE}(\widehat{f}) = \frac{1}{n} \sum_{i=1}^{n} (f(\boldsymbol{x_i}) - \widehat{f}(\boldsymbol{x_i}))^2 \ .$$

Throughout this paper, we let $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ be the *data matrix*, i.e., the matrix whose j-th row is $\boldsymbol{x}_j^T$ for every j, and we let r denote the rank of $\boldsymbol{X}$.

The following notation will also be useful. For any function $f : \mathbb{R}^d \to \mathbb{R}$, we let $\boldsymbol{f} \in \mathbb{R}^n$ denote the vector with components $\boldsymbol{f}_i = f(\boldsymbol{x}_i)$ for $i \in [n]$. For any interval I, we let $\boldsymbol{X}^I$ denote the data matrix consisting of all data points $\boldsymbol{x}_i$ for $i \in I$, and for any vector $\boldsymbol{v} \in \mathbb{R}^n$, we let $\boldsymbol{v}^I \in \mathbb{R}^{|I|}$ be the vector of $\boldsymbol{v}_i$ for $i \in I$.

2.1. Our Contributions

Our main contributions are new, fast algorithms for the aforementioned segmented regression problem. We now informally state our main results and refer to later sections

for more precise theorems.

Theorem 2 (informal statement of Theorems 13 and 14)**.** *There is an algorithm* GREEDYMERGE*, which, given $\boldsymbol{X}$ (of rank r), $\boldsymbol{y}$, a target number of pieces k, and the variance of the noise s^2, runs in time $O(nd^2 \log n)$ and outputs an $O(k)$-piecewise linear function $\widehat{f}$ so that with probability at least 0.99, we have*

$$\mathrm{MSE}(\widehat{f}) \;\leq\; O\left(\sigma^2 \frac{kr}{n} + \sigma\sqrt{\frac{k}{n}}\log n\right) .$$

That is, our algorithm runs in time which is *nearly linear* in n and still achieves a reasonable rate of convergence. While this rate is asymptotically slower than the rate of the dynamic programming estimator, our algorithm is significantly faster than the DP so that in order to achieve a comparable MSE, our algorithm takes less total time given access to a sufficient number of samples. For more details on this comparision and an experimental evaluation, see Sections 2.2 and 4.

At a high level, our algorithm first forms a fine partition of $[n]$ and then iteratively merges pieces of this partitions until only $O(k)$ pieces are left. In each iteration, the algorithm reduces the number of pieces in the following manner: we group consecutive intervals into pairs which we call "candidates". For each candidate interval, the algorithm computes an error term that is the error of a least squares fit combined with a regularizer depending on the variance of the noise s^2. The algorithm then finds the $O(k)$ candidates with the largest errors. We do not merge these candidates, but do merge all other candidate intervals. We repeat this process until only $O(k)$ pieces are left.

A drawback of this algorithm is that we need to know the variance of the noise s^2, or at least have a good estimate. In practice, we might be able to obtain such an estimate, but ideally our algorithm would work without knowing s^2. By extending our algorithm, we obtain the following result:

Theorem 3 (informal)**.** *There is an algorithm* BUCKETGREEDYMERGE*, which, given $\boldsymbol{X}$ (of rank r), $\boldsymbol{y}$, and a target number of pieces k, runs in time $O(nd^2 \log n)$ and outputs an $O(k \log n)$-piecewise linear function $\widehat{f}$ so that with probability at least 0.99, we have*

$$\mathrm{MSE}(\widehat{f}) \;\leq\; O\left(\sigma^2 \frac{kr \log n}{n} + \sigma\sqrt{\frac{k}{n}}\log n\right) .$$

At a high level, there are two fundamental changes to the algorithm: first, instead of merging with respect to the sum squared error of the least squares fit regularized by a term depending on s^2, we instead merge with respect to the average error the least squares fit incurs within the current interval. The second change is more substantial: instead of finding the $O(k)$ candidates with largest error and merging the rest, we now split the candidates into $\log n$ buckets based on the lengths of the candidate intervals. In this scheme, bucket α contains all candidates with length between 2^α and $2^{\alpha+1}$, for $\alpha = 0, \ldots, \log n - 1$. Then we find the $k+1$ candidates with largest error within each bucket and merge the remaining candidate intervals. We continue this process until we are left with $O(k \log n)$ buckets.

A potential disadvantage of our algorithms above is that they produce $O(k)$ or $O(k \log n)$ pieces, respectively. In order to address this issue, we also provide a postprocessing algorithm that converts the output of any of our algorithms and decreases the number of pieces down to $2k+1$ while preserving the statistical guarantees above. The guarantee of this postprocessing algorithm is as follows.

Theorem 4 (informal)**.** *There is an algorithm* POSTPROCESSING *that takes as input the output of either* GREEDYMERGE *or* BUCKETGREEDYMERGE *together with a target number of pieces k, runs in time $O\left(k^3 d^2 \log n\right)$, and outputs a $(2k+1)$-piecewise linear function $\widehat{f^p}$ so that with probability at least 0.99, we have*

$$\mathrm{MSE}(\widehat{f^p}) \;\leq\; O\left(\sigma^2 \frac{kr}{n} + \sigma\sqrt{\frac{k}{n}}\log n\right) .$$

Qualitatively, an important aspect this algorithm is that its running time depends only logarithmically on n. In practice, we expect k to be much smaller than n, and hence the running time of this postprocessing step will usually be dominated by the running time of the piecewise linear fitting algorithm run before it.

In this document, we only give the formal pseudo code and proofs for GREEDYMERGE. We refer the reader to the full version for more details.

2.2. Comparison to prior work

Dynamic programming. Previous work on segmented regression with statistical guarantees (Bai & Perron, 1998; Yamamoto & Perron, 2013) relies heavily on dynamic programming-style algorithms to find the k-piecewise linear least-squares estimator. Somewhat surprisingly, we are not aware of any work which explicitly gives the best possible running time and statistical guarantee for this algorithm. For completeness, we prove the following result (Theorem 5), which we believe to be folklore. The techniques used in its proof will also be useful for us later.

Theorem 5 (informal)**.** *The exact dynamic program runs in time $O(n^2(d^2+k))$ and outputs an k-piecewise linear estimator $\widehat{f}$ so that with probability at least 0.99 we have* $\mathrm{MSE}(\widehat{f}) \;\leq\; O\left(\sigma^2 \frac{kr}{n}\right)$.

We now compare our guarantees with those of the DP. The main advantage of our approaches is computational

efficiency – our algorithm runs in linear time, while the running time of the DP has a quadratic dependence on n. While our statistical rate of convergence is slower, we are able to achieve the same MSE as the DP in asymptotically less time if enough samples are available.

For instance, suppose we wish to obtain a MSE of η with a k-piecewise linear function, and suppose for simplicity that $d = O(1)$ so that $r = O(1)$ as well. Then Theorem 5 tells us that the DP requires $n = k/\eta$ samples and runs in time $O(k^3/\eta^2)$. On the other hand, Theorem 2 tells us that GREEDYMERGING requires $n = \widetilde{O}(k/\eta^2)$ samples and thus runs in time $\widetilde{O}(k/\eta^2)$. For non-trivial values of k, this is a significant improvement in time complexity.

This gap also manifests itself strongly in our experiments (see Section 4). When given the same number of samples, our MSE is a factor of 2-4 times worse than that achieved by the DP, but our algorithm is about $1,000$ times faster already for 10^4 data points. When more samples are available for our algorithm, it achieves the same MSE as the DP about 100 times faster.

Histogram Approximation Our results build upon the techniques of (Acharya et al., 2015), who consider the problem of *histogram approximation*. Indeed, without too much work it is possible to convert their algorithm to our regression setting, but the resulting guarantee is not statistically meaningful (see the full version for a more detailed discussion). To overcome this difficulty, we propose a merging algorithm based on a new merging error measure. Utilizing more sophisticated proof techniques than in (Acharya et al., 2015), we show that our new algorithm has good statistical guarantees in the regression setting.

2.3. Mathematical Preliminaries

Tail bounds We will require the following tail bound on deviations of squares of sub-Gaussian random variables. We defer the proof to the supplementary material.

Corollary 6. *Fix $\delta > 0$ and let $\epsilon_1, \ldots, \epsilon_n$ be as in (1). Recall that $s = \mathbb{E}[\epsilon_i^2]$. Then, with probability $1-\delta$, we have that simultaneously for all intervals $I \subseteq [n]$ the following inequality holds:*

$$\left| \sum_{i \in I} \epsilon_i^2 - s^2 |I| \right| \leq O(\sigma^2 \cdot \log(n/\delta)) \sqrt{|I|} \,. \qquad (2)$$

Linear regression. Our analysis builds on the classical results for fixed design linear regression. In linear regression, the generative model is exactly of the form described in (1), except that f is restricted to be a 1-piecewise linear function (as opposed to a k-piecewise linear function), i.e., $f(\boldsymbol{x}) = \langle \boldsymbol{\theta}^*, \boldsymbol{x} \rangle$ for some unknown $\boldsymbol{\theta}^*$.

The problem of linear regression is very well understood, and the asymptotically best estimator is known to be the *least-squares* estimator.

Definition 7. *Given $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and $y_1, \ldots, y_n$, the* least squares estimator f^{LS} *is defined to be the linear function which minimizes $\sum_{i=1}^n (y_i - f(\boldsymbol{x}_i))^2$ over all linear functions f. For any interval I, we let f_I^{LS} denote the least squares estimator for the data points restricted to I, i.e. for the data pairs $\{(\boldsymbol{x}_i, y_i)\}_{i \in I}$. We also let* LEASTSQUARES *$(\boldsymbol{X}, \boldsymbol{y}, I)$ denote an algorithm which solves linear least squares for these data points, i.e., which outputs the coefficients of the linear least squares fit for these points.*

Following our previous definitions, we let $\boldsymbol{f}^{\mathrm{LS}} \in \mathbb{R}^n$ denote the vector whose i-th coordinate is $f^{\mathrm{LS}}(\boldsymbol{x}_i)$, and similarly for any $I \subseteq [n]$ we let $\boldsymbol{f}_I^{\mathrm{LS}} \in \mathbb{R}^{|I|}$ denote the vector whose i-th coordinate for $i \in I$ is $f_I^{\mathrm{LS}}(\boldsymbol{x}_i)$. The following error rate is known for the least-squares estimator:

Fact 8. *Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and $y_1, \ldots, y_n$ be generated as in (1), where f is a linear function. Let $f^{\mathrm{LS}}(x)$ be the least squares estimator. Then,*

$$\mathbb{E}\left[\mathrm{MSE}(f^{\mathrm{LS}})\right] = O\left(\sigma^2 \frac{r}{n}\right) .$$

Moreover, with probability $1 - \delta$, we have

$$\mathrm{MSE}(f^{\mathrm{LS}}) = O\left(\sigma^2 \frac{r + \log 1/\delta}{n}\right) .$$

Fact 8 can be proved with the following lemma, which we also use in our analysis. The lemma bounds the correlation of a random vector with any fixed r-dimensional subspace:

Lemma 9 (c.f. proof of Theorem 2.2 in (Rigollet, 2015))**.** *Fix $\delta > 0$. Let $\epsilon_1, \ldots, \epsilon_n$ be i.i.d. sub-Gaussian random variables with variance proxy σ^2. Let $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$, and let S be a fixed r-dimensional affine subspace of $\mathbb{R}^n$. Then, with probability $1 - \delta$, we have*

$$\sup_{\boldsymbol{v} \in S \setminus \{0\}} \frac{|\boldsymbol{v}^T \boldsymbol{\epsilon}|}{\|\boldsymbol{v}\|} \leq O\left(\sigma \sqrt{r + \log(1/\delta)}\right) .$$

This lemma also yields the two following consequences. The first corollary bounds the correlation between sub-Gaussian random noise and any linear function on any interval (see the full paper for proofs):

Corollary 10. *Fix $\delta > 0$ and $\boldsymbol{v} \in \mathbb{R}^n$. Let $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ be as in Lemma 9. Then with probability at least $1 - \delta$, we have that for all intervals I, and for all non-zero linear functions $f : \mathbb{R}^d \to \mathbb{R}$,*

$$\frac{|\langle \boldsymbol{\epsilon}^I, \boldsymbol{f}_I + \boldsymbol{v}_I \rangle|}{\|\boldsymbol{f}_I + \boldsymbol{v}_I\|} \leq O(\sigma \sqrt{r + \log(n/\delta)}) \,.$$

Corollary 11. *Fix $\delta > 0$ and $\boldsymbol{v} \in \mathbb{R}^n$. Let $\boldsymbol{\epsilon}$ be as in*

Lemma 9. Then with probability at least $1-\delta$, we have

$$\sup_{f\in\mathcal{L}_k} \frac{|\langle \boldsymbol{\epsilon}, \boldsymbol{f}_I + \boldsymbol{v}_I\rangle|}{\|\boldsymbol{f}_I + \boldsymbol{v}_I\|} \;\le\; O\left(\sigma\sqrt{kr + k\log\frac{n}{\delta}}\right) .$$

These bounds also imply the following statement, which we will need later. We defer the proof to the supplementary material.

Lemma 12. *Fix $\delta > 0$. Then with probability $1-\delta$ we have the following: for all disjoint sets of k intervals $I_1, \ldots, I_k$ of $[n]$ so that f is flat on each I_ℓ, the following inequality holds:*

$$\sum_{\ell=1}^{k} \|\boldsymbol{f}^{\mathrm{LS}}_{I_\ell} - \boldsymbol{f}_{I_\ell}\|^2 \;\le\; O(\sigma^2\, k(r + \log(n/\delta))) .$$

3. A greedy merging algorithm

In this section, we formally state our new algorithm for segmented regression and analyze its statistical and computational efficiency. See Algorithm 1 for the corresponding pseudocode. The algorithm expects two tuning parameters τ, γ because the algorithm does not return a k-piecewise linear function but instead a $O(k)$-piecewise linear function The tuning parameters allow us to trade off running time for fewer pieces. In typical use cases we have $\tau, \gamma = \Theta(1)$, in which case our algorithm produces an $O(k)$-piecewise linear function in time $O(nd^2 \log n)$ time.

We now state the running time of GREEDYMERGING. The analysis is similar to the analysis presented in (Acharya et al., 2015) and for space considerations is left to the supplementary material.

Theorem 13. *Let $\boldsymbol{X}$ and $\boldsymbol{y}$ be as above. Then* GREEDYMERGING$(\tau, \gamma, s, \boldsymbol{X}, \boldsymbol{y})$ *outputs a $\big((2+\frac{2}{\tau})k + \gamma\big)$-piecewise linear function and runs in time $O(nd^2 \log(n/\gamma))$.*

3.1. Analysis of GREEDYMERGING

Theorem 14. *Let $\delta > 0$, and let $\widehat{f}$ be the estimator returned by* GREEDYMERGING. *Let $m = (2+\frac{2}{\tau})k + \gamma$ be the number of pieces in $\widehat{f}$. Then, with probability $1-\delta$, we have that*

$$\mathrm{MSE}(\widehat{f}) \le O\left(\sigma^2\left(\frac{m(r+\log(n/\delta))}{n}\right) + \sigma\frac{\tau+\sqrt{k}}{\sqrt{n}}\log\left(\frac{n}{\delta}\right)\right) .$$

Proof. We first condition on the event that Corollaries 6, 10 and 11, and Lemma 12 all hold with error parameter $O(\delta)$, so that together they all hold with probability at least $1-\delta$. Let $\mathcal{I} = \{I_1, \ldots, I_m\}$ be the final partition of $[n]$ that our algorithm produces. Recall f is the ground truth k-piecewise linear function. We partition the intervals in $\mathcal{I}$ into two sets:

$$\mathcal{F} = \{I \in \mathcal{I} : f \text{ is flat on } I\} ,$$
$$\mathcal{J} = \{I \in \mathcal{I} : f \text{ has a jump on } I\} .$$

We first bound the error over the intervals in $\mathcal{F}$. By Lemma 12, we have

$$\sum_{I\in\mathcal{F}} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; O(\sigma^2|\mathcal{F}|(r + \log(n/\delta))) , \qquad (3)$$

with probability at least $1 - O(\delta)$.

We now turn our attention to the intervals in $\mathcal{J}$ and distinguish two further cases. We let $\mathcal{J}_1$ be the set of intervals in $\mathcal{J}$ which were never merged, and we let $\mathcal{J}_2$ be the remaining intervals. If the interval $I \in \mathcal{J}_1$ was never merged, the interval contains one point, call it i. Because we may assume that $\boldsymbol{x}_i \neq 0$, we know that for this one point, our estimator satisfies $\widehat{f}(\boldsymbol{x}_i) = y_i$, since this is clearly the least squares fit for a linear estimator on one nonzero point. Hence Corollary 6 implies that the following inequality holds with probability at least $1 - O(\delta)$:

$$\begin{aligned}\sum_{I\in\mathcal{J}_1} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 &= \sum_{I\in\mathcal{J}_1} \|\epsilon_I\|^2 \\ &\le \sigma^2\left(\sum_{I\in\mathcal{J}_1}|I| + O\left(\log\frac{n}{\delta}\right)\sqrt{\sum_{I\in\mathcal{J}_1}|I|}\right) \\ &\le \sigma^2\left(m + O\left(\log\frac{n}{\delta}\right)\sqrt{m}\right) . \qquad (4)\end{aligned}$$

We now finally turn our attention to the intervals in $\mathcal{J}_2$. Fix an interval $I \in \mathcal{J}_2$. By definition, the interval I was merged in some iteration of the algorithm. This implies that in that iteration, there were $(1+1/\tau)k$ intervals $M_1, \ldots, M_{(1+1/\tau)k}$ so that for each interval M_ℓ, we have

$$\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| \;\le\; \|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\|^2 - s^2|M_\ell| . \qquad (5)$$

Since the true, underlying k-piecewise linear function f has at most k jumps, this means that there are at least k/τ intervals of the M_ℓ on which f is flat. WLOG assume that these intervals are $M_1, \ldots, M_{k/\tau}$.

We have, by definition,

$$\begin{aligned}\sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 - s^2|M_\ell| &= \sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 \\ &+ 2\sum_{\ell=1}^{k/\tau}\langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\rangle + \sum_{\ell=1}^{k/\tau}\sum_{i\in M_\ell}(\epsilon_i^2 - s^2) . \qquad (6)\end{aligned}$$

We will upper bound each term on the RHS in turn. First, since the function f is flat on each M_ℓ for $\ell = 1, \ldots, k/\tau$, Lemma 12 implies

$$\sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 \le O\left(\sigma^2\frac{k}{\tau}\left(r + \log\frac{n}{\delta}\right)\right) , \qquad (7)$$

Algorithm 1 Piecewise linear regression by greedy merging.

GREEDYMERGING($\tau, \gamma, s, \boldsymbol{X}, \boldsymbol{y}$)
{*Initial partition of* $[n]$ *into intervals of length* 1.}
$\mathcal{I}^0 \leftarrow \{\{1\}, \{2\}, \ldots, \{n\}\}$
{*Iterative greedy merging (we start with* $j \leftarrow 0$*).*}
while $|\mathcal{I}^j| > (2 + \frac{2}{\tau})k + \gamma$ **do**
  Let s_j be the current number of intervals. {*Compute the least squares fit and its error for merging neighboring pairs of intervals.*}
  for $u \in \{1, 2, \ldots, \frac{s_j}{2}\}$ **do**
    $\theta_u \leftarrow$ LEASTSQUARES$(\boldsymbol{X}, \boldsymbol{y}, I_{2u-1} \cup I_{2u})$
    $e_u = \|\boldsymbol{y}^I - \boldsymbol{X}^I \theta_u\|_2^2 - s^2 |I_{2u-1} \cup I_{2u}|$
  end for
  Let L be the set of indices u with the $(1 + \frac{1}{\tau})k$ largest errors e_u, breaking ties arbitrarily.
  Let M be the set of the remaining indices. {*Keep the intervals with large merging errors.*}
  $\mathcal{I}^{j+1} \leftarrow \bigcup_{u \in L} \{I_{2u-1}, I_{2u}\}$ {*Merge the remaining intervals.*}
  $\mathcal{I}^{j+1} \leftarrow \mathcal{I}^{j+1} \cup \{I_{2u-1} \cup I_{2u} \,|\, u \in M\}$
  $j \leftarrow j + 1$
end while
return the least squares fit to the data on every interval in $\mathcal{I}^j$

with probability at least $1 - O(\delta)$.

Moreover, note that the function $f^{\mathrm{LS}}_{M_\ell}$ is a linear function on M_ℓ of the form $f^{\mathrm{LS}}_{M_\ell}(\boldsymbol{x}) = \boldsymbol{x}^T \hat{\beta}$, where $\hat{\beta} \in \mathbb{R}^d$ is the least-squares fit on M_ℓ. Because $\boldsymbol{f}$ is just a fixed vector, the vector $\boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}$ lives in the affine subspace of vectors of the form $\boldsymbol{f}_{M_\ell} + (\boldsymbol{X}_{M_\ell})\eta$ where $\eta \in \mathbb{R}^d$ is arbitrary. So Corollary 11 and (7) imply that

$$\begin{aligned}
&\sum_{\ell=1}^{k/\tau} \langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell} \rangle \\
&\le \sqrt{\sum_{\ell=1}^{k/\tau} \langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell} \rangle} \cdot \sup_{\eta} \frac{|\langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{X}\eta \rangle|}{\|\boldsymbol{X}\eta\|} \\
&\le O\left(\sigma^2 \frac{k}{\tau}\left(r + \log\frac{n}{\delta}\right)\right) . \qquad (8)
\end{aligned}$$

with probability $1 - O(\delta)$.

By Corollary 6, we get that with probability $1 - O(\delta)$,

$$\sum_{\ell=1}^{k/\tau} \left(\sum_{i \in M_\ell} \epsilon_i^2 - s^2 |M_\ell| \right) \le O\left(\sigma \log\frac{n}{\delta}\right) \sqrt{n} .$$

Putting it all together, we get that

$$\begin{aligned}
&\sum_{i=1}^{k/\tau} \left(\left\| \boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell} \right\|^2 - s^2 |M_\ell| \right) \\
&\le O\left(\frac{k}{\tau}\sigma^2\left(r + \log\frac{n}{\delta}\right)\right) + O\left(\sigma \log\frac{n}{\delta}\right)\sqrt{n} \qquad (9)
\end{aligned}$$

with probability $1 - O(\delta)$. Since the LHS of (5) is bounded by each individual summand above, this implies that the LHS is also bounded by their average, which implies that

$$\begin{aligned}
&\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2 |I| \\
&\le \frac{\tau}{k} \sum_{i=1}^{k/\tau} \left(\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\|^2 - s^2 |M_\ell| \right) \\
&\le O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) + O\left(\sigma \log\frac{n}{\delta}\right) \frac{\tau\sqrt{n}}{k} . \qquad (10)
\end{aligned}$$

We now similarly expand out the LHS of (5):

$$\begin{aligned}
&\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2 |I| \\
&= \|\boldsymbol{f}_I + \boldsymbol{\epsilon}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2 |I| \\
&= \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 + 2\langle \boldsymbol{\epsilon}_I, \boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I \rangle + \|\boldsymbol{\epsilon}_I\|^2 - s^2 |I| . \qquad (11)
\end{aligned}$$

Next, we aim to upper bound $\sum_{i \in I} (f(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i))^2$. Hence, we lower bound the second and third terms of (11). The calculations here closely mirror those above.

By Corollary 10, we have that

$$2\langle \boldsymbol{\epsilon}_I, \boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I \rangle \;\ge\; -O\left(\sigma\sqrt{r + \log\left(\frac{n}{\delta}\right)}\right) \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\| ,$$

and by Corollary 6 we have

$$\|\boldsymbol{\epsilon}_I\|^2 - s^2 |I| \;\ge\; -O\left(\sigma \log\frac{n}{\delta}\right)\sqrt{|I|} ,$$

and so

$$\begin{aligned}
&\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2 |I| \\
&\ge \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 - O\left(\sigma\sqrt{r + \log\left(\frac{n}{\delta}\right)}\right) \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\| \\
&\quad - O\left(\sigma \log\frac{n}{\delta}\right)\sqrt{|I|} . \qquad (12)
\end{aligned}$$

Putting (10) and (12) together yields that with probability $1 - O(\delta)$,

$$\begin{aligned}\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; & O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) \\ & + O\left(\sigma\sqrt{r + \log\left(\frac{n}{\delta}\right)}\right)\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\| \\ & + O\left(\sigma \log\frac{n}{\delta}\right)\left(\frac{\tau\sqrt{n}}{k} + \sqrt{|I|}\right) .\end{aligned}$$

Letting $z^2 = \|\widehat{\boldsymbol{f}}_I - \boldsymbol{f}_I\|^2$, then this inequality is of the form $z^2 \le bz + c$ where $b, c \ge 0$. In this case, we have that

$$b = O\left(\sigma\sqrt{r + \log\frac{n}{\delta}}\right) \text{ , and}$$

$$c = O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) + O\left(\sigma\log\frac{n}{\delta}\right)\left(\frac{\tau\sqrt{n}}{k} + \sqrt{|I|}\right)$$

We now prove the following lemma about the behavior of such quadratic inequalities:

Lemma 15. *Suppose $z^2 \le bz + c$ where $b, c \ge 0$. Then $z^2 \le O(b^2 + c)$.*

Proof. From the quadratic formula, the inequality implies that $z \le \frac{b + \sqrt{b^2 + 4c}}{2}$. From this, it is straightforward to demonstrate the desired claim. □

Thus, from the lemma, we have

$$\begin{aligned}\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; & O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) \\ & + O\left(\sigma\log\frac{n}{\delta}\right)\left(\frac{\tau\sqrt{n}}{k} + \sqrt{|I|}\right) .\end{aligned}$$

Hence the total error over all intervals in $\mathcal{J}_2$ can be bounded by:

$$\begin{aligned}\sum_{I\in\mathcal{J}_2}\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; & O\left(k\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) \\ & + O\left(\sigma\log\frac{n}{\delta}\right)\left(\tau\sqrt{n} + \sum_{I\in\mathcal{J}}\sqrt{|I|}\right) \\ \le\; & O\left(k\sigma^2\left(r + \log\left(\frac{n}{\delta'}\right)\right)\right) \\ & + O\left(\sigma\log\frac{n}{\delta}\right)\left(\tau\sqrt{n} + \sqrt{kn}\right) . \qquad (13)\end{aligned}$$

In the last line we use that the intervals $I \in \mathcal{J}_2$ are disjoint (and hence their cardinalities sum up to at most n), and that there are at most k intervals in $\mathcal{J}_2$ because the function f is k-piecewise linear. Finally, applying a union bound and summing (3), (4), and (13) yields the desired conclusion. □

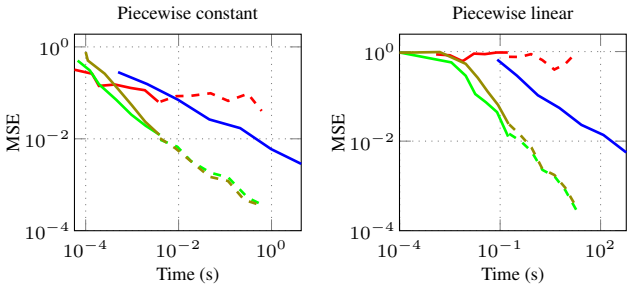

Figure 2. Computational vs. statistical efficiency in the synthetic data experiments. The solid lines correspond to the data in Figure 1, the dashed lines show the results from additional runs of the merging algorithms for larger values of n. The merging algorithm achieves the same MSE as the dynamic program about $100\times$ faster if a sufficient number of samples is available.

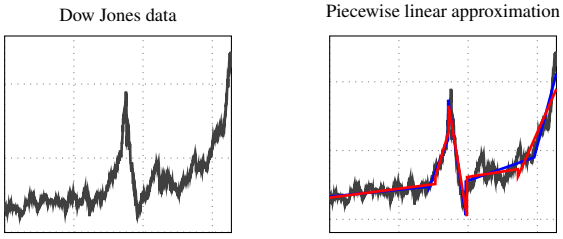

Figure 3. Results of fitting a 5-piecewise linear function ($d = 2$) to a Dow Jones time series. The merging algorithm produces a fit that is comparable to the dynamic program and is about $200\times$ faster (0.013 vs. 3.2 seconds).

4. Experiments

In addition to our theoretical analysis above, we also study the empirical performance of our new estimator for segmented regression on both real and synthetic data. As baseline, we compare our estimator (GREEDYMERGING) to the dynamic programming approach. Since our algorithm combines both combinatorial and linear-algebraic operations, we use the Julia programming language[1] (version 0.4.2) for our experiments because Julia achieves performance close to C on both types of operations. All experiments were conducted on a laptop computer with a 2.8 GHz Intel Core i7 CPU and 16 GB of RAM.

Synthetic data. Experiments with synthetic data allow us to study the statistical and computational performance of our estimator as a function of the problem size n. Our theoretical bounds indicate that the worst-case performance of the merging algorithm scales as $O(\frac{kd}{n} + \sqrt{k/n}\log n)$ for constant error variance. Compared to the $O(\frac{kd}{n})$ rate of the dynamic program, this indicates that the relative performance of our algorithm can depend on the number of features d. Hence we use two types of synthetic data: a piecewise-constant function f (effectively $d = 1$) and a

[1] http://julialang.org/

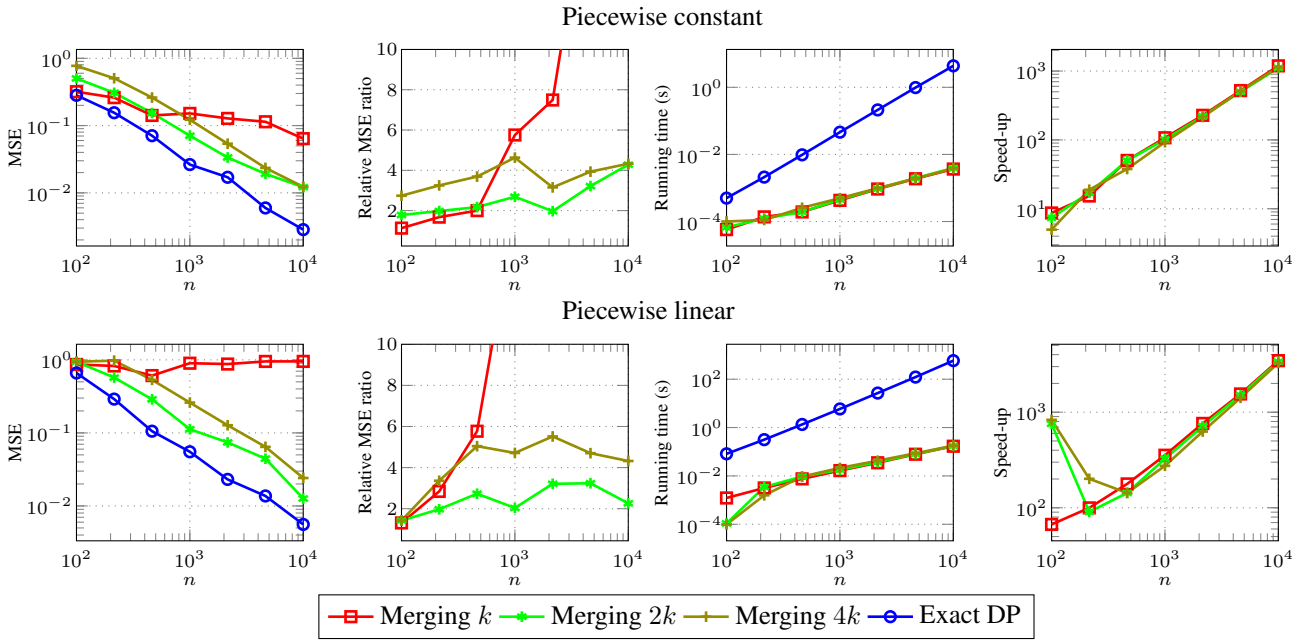

Figure 1. Experiments with synthetic data: results for piecewise constant models with $k = 10$ segments (top row) and piecewise linear models with $k = 5$ segments (bottom row, dimension $d = 10$). Compared to the exact dynamic program, the MSE achieved by the merging algorithm is worse but stays within a factor of 2 to 4 for a sufficient number of output segments. The merging algorithm is significantly faster and achieves a speed-up of about $10^3\times$ compared to the dynamic program for $n = 10^4$. This leads to a significantly better trade-off between statistical and computational performance (see also Figure 2).

piecewise linear function f with $d = 10$ features.

We generate the piecewise constant function f by randomly choosing $k = 10$ integers from the set $\{1, \dots, 10\}$ as function value in each of the k segments.[2] Then we draw n/k samples from each segment by adding an i.i.d. Gaussian noise term with variance 1 to each sample.

For the piecewise linear case, we generate a $n \times d$ data matrix $\boldsymbol{X}$ with i.i.d. Gaussian entries ($d = 10$). In each segment I, we choose the parameter values $\boldsymbol{\beta}_I$ independently and uniformly at random from the interval $[-1, 1]$. So the true function values in this segment are given by $\boldsymbol{f}_I = \boldsymbol{X}_I \boldsymbol{\beta}_I$. As before, we then add an i.i.d. Gaussian noise term with variance 1 to each function value.

Figure 1 shows the results of the merging algorithm and the exact dynamic program for sample size n ranging from 10^2 to 10^4. Since the merging algorithm can produce a variable number of output segments, we run the merging algorithm with three different parameter settings corresponding to k, $2k$, and $4k$ output segments, respectively. As predicted by our theory, the plots show that the exact dynamic program has a better statistical performance. However, the MSE of the merging algorithm with $2k$ pieces is only worse by a factor of 2 to 4, and this ratio empirically increases only slowly with n (if at all). The experiments also show that forcing the merging algorithm to return at most k pieces can lead to a significantly worse MSE.

In terms of computational performance, the merging algorithm has a significantly faster running time, with speed-ups of more than $1,000\times$ for $n = 10^4$ samples. As can be seen in Figure 2, this combination of statistical and computational performance leads to a significantly improved trade-off between the two quantities. When we have a sufficient number of samples, the merging algorithm achieves a given MSE roughly $100\times$ faster than the dynamic program.

Real data. We also investigate whether the merging algorithm can empirically be used to find linear trends in a real dataset. We use a time series of the Dow Jones index as input, and fit a piecewise linear function ($d = 2$) with 5 segments using both the dynamic program and our merging algorithm with $k = 5$ output pieces. As can be seen from Figure 3, the dynamic program produces a slightly better fit for the rightmost part of the curve, but the merging algorithm identifies roughly the same five main segments. As before, the merging algorithm is significantly faster and achieves a $200\times$ speed-up compared to the dynamic program (0.013 vs 3.2 seconds).

[2] We also repeated the experiment for other values of k. Since the results are not qualitatively different, we only report the $k = 10$ case here.

Acknowledgements

Part of this research was conducted while Ilias Diakonikolas was at the University of Edinburgh, Jerry Li was at Microsoft Research Cambridge (UK), and Ludwig Schmidt was at UC Berkeley.

Jayadev Acharya was supported by a grant from the MIT-Shell Energy Initiative. Ilias Diakonikolas was supported in part by EPSRC grant EP/L021749/1, a Marie Curie Career Integration Grant, and a SICSA grant. Jerry Li was supported by NSF grant CCF-1217921, DOE grant DE-SC0008923, NSF CAREER Award CCF-145326, and a NSF Graduate Research Fellowship. Ludwig Schmidt was supported by grants from the MIT-Shell Energy Initiative, MADALGO, and the Simons Foundation.

References

Acharya, J., Diakonikolas, I., Hegde, C., Li, J. Z., and Schmidt, L. Fast and near-optimal algorithms for approximating distributions by histograms. In *PODS*, pp. 249–263, 2015a.

Acharya, J., Diakonikolas, I., Li, J. Zheng, and Schmidt, L. Sample-optimal density estimation in nearly-linear time. *CoRR*, abs/1506.00671, 2015b. URL http://arxiv.org/abs/1506.00671.

Avron, H., Sindhwani, V., and Woodruff, D. Sketching structured matrices for faster nonlinear regression. In *NIPS*, pp. 2994–3002. 2013.

Bai, J. and Perron, P. Estimating and testing linear models with multiple structural changes. *Econometrica*, 66(1): 47–78, 1998.

Chatterjee, S., Guntuboyina, A., and Sen, B. On risk bounds in isotonic and other shape restricted regression problems. *Annals of Statistics*, 43(4):1774–1800, 08 2015.

Feder, P. I. On asymptotic distribution theory in segmented regression problems– identified case. *Annals of Statistics*, 3(1):49–83, 01 1975.

Friedman, J. H. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67, 03 1991.

Gallant, A. R. and A., Fuller W. Fitting segmented polynomial regression models whose join points have to be estimated. *Journal of the American Statistical Association*, 68(341):144–147, 1973.

Jordan, M. I. On statistics, computation and scalability. *Bernoulli*, 19(4):1378–1390, 09 2013.

Kyng, R., Rao, A., and Sachdeva, S. Fast, provable algorithms for isotonic regression in all l_p-norms. In *NIPS*, pp. 2701–2709, 2015.

Meyer, M. C. Inference using shape-restricted regression splines. *Annals of Applied Statistics*, 2(3):1013–1033, 09 2008.

Mosteller, F. and Tukey, J. W. *Data analysis and regression: a second course in statistics*. Addison-Wesley, Reading (Mass.), Menlo Park (Calif.), London, 1977.

Rigollet, P. High dimensional statistics. 2015. URL http://www-math.mit.edu/~rigollet/PDFs/RigNotes15.pdf.

Stone, C. J. The use of polynomial splines and their tensor products in multivariate function estimation. *Annals of Statistics*, 22(1):pp. 118–171, 1994.

Stone, C. J., Hansen, M. H., Kooperberg, C., and Truong, Y. K. Polynomial splines and their tensor products in extended linear modeling: 1994 wald memorial lecture. *Annals of Statistics*, 25(4):1371–1470, 1997.

Wegman, E. J. and Wright, I. W. Splines in statistics. *Journal of the American Statistical Association*, 78(382):pp. 351–365, 1983.

Yamamoto, Y. and Perron, P. Estimating and testing multiple structural changes in linear models using band spectral regressions. *Econometrics Journal*, 16(3):400–429, 2013.