Fast Algorithms for Segmented Regression

Jayadev Acharya
MIT
jayadev@csail.mit.edu

Ilias Diakonikolas
University of Southern California
ilias.d@ed.ac.uk

Jerry Li
MIT
jerryzli@csail.mit.edu

Ludwig Schmidt
MIT
ludwigs@mit.edu

June 20, 2016

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., [MT77] 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., [GA73, Fed75, Fri91, BP98, YP13, KRS15, ASW13, Mey08, CGS15]). 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 [GA73, Fed75, BP98, YP13]. 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., [WW83, Fri91, Sto94, SHKT97, Mey08].

Information-theoretic aspects of the segmented regression problem are well-understood: Roughly speaking, the minimax 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 [BP98, YP13] 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 [Jor13]: "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 [ADH+15, ADLS15], 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.[1] 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.*

Later in the paper (see Section 6), we also discuss the 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.

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 $\boldsymbol{X}$. We also assume that no $\boldsymbol{x}_i$ is the all-zeros vector, since such points are trivially fit by any linear function.

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 $i \in I$.

[1] We observe that s^2 is guaranteed to be finite since the ϵ_i are sub-Gaussian. The variance s^2 is in general not equal to the variance proxy σ^2, however, it is well-known that $s^2 \leq \sigma^2$.

We remark that this model also contains the problem of (fixed design) piecewise polynomial regression as an important subcase. Indeed, imagine we are given $x_1, \ldots, x_n \in \mathbb{R}$ and $y_1, \ldots, y_n \in \mathbb{R}$ so that there is some k-piece degree-d polynomial p so that for all $i = 1, \ldots, n$ we have $y_i = p(x_i) + \epsilon_i$, and our goal is to recover p in the same mean-squared sense as above. We can write this problem as a k-piecewise linear fit in $\mathbb{R}^{d+1}$, by associating the data vector $\boldsymbol{v}_i = (1, x_i, x_i^2, \ldots, x_i^d)$ to each x_i. If the piecewise polynomial p has breakpoints at $z_1, \ldots z_{k-1}$, the associated partition for the k-piecewise linear function is $\mathcal{I} = \{(-\infty, z_1), [z_1, z_2), \ldots, (z_{k-2}, z_{k-1}], (z_{k-1}, \infty)\}$. If the piecewise polynomial is of the form $p(x) = \sum_{\ell=0}^{d} a_\ell^I x^\ell$ for any interval $I \in \mathcal{I}$, then for any vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_{d+1}) \in \mathbb{R}^{d+1}$, the ground truth k-piecewise linear function is simply the linear function $f(\boldsymbol{v}) = \sum_{\ell=1}^{d+1} a_{\ell-1}^I v_\ell$ for $v_2 \in I$. Moreover, the data matrix associated with the data points $\boldsymbol{v}$ is a Vandermonde matrix. For $n \geq d+1$ it is well-known that this associated Vandermonde matrix has rank exactly $d+1$ (assuming $x_i \neq x_j$ for any $i \neq j$).

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 18 and 19)**.** *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 7.

At a high level, our algorithm proceeds as follows: it 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 of it. In practice, we might be able to obtain such an estimate, but ideally our algorithm would work without knowing s^2. By extending our greedy algorithm, we obtain the following result:

Theorem 3 (informal statement of Theorems 21 and 22)**.** *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 top $O(k)$ 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 bucketing 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. Intuitively, this bucketing allows us to control the variance of the noise without knowing s^2 because all candidate intervals have roughly the same length.

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 statement of Theorems 23 and 24)**.** *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.

2.2 Comparison to prior work

Dynamic programming. Previous work on segmented regression with statistical guarantees [BP98, YP13] 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 statement of Theorems 15 and 16)**.** *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 (and also in practice) 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 (ignoring log factors) 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 7). 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 [ADH$^+$15], who consider the problem of *histogram approximation*. In this setting, one is given a function $f : [n] \to \mathbb{R}$, and the goal is to find the best k-flat approximation to f in sum-squared error. This problem bears nontrivial resemblance to the problem of piecewise constant regression, and indeed, the results in this paper establish a connection between the two problems. The histogram approximation problem has received significant attention in the database community (e.g. [JKM$^+$98, GKS06, ADH$^+$15]), and it is natural to ask whether these results also imply algorithms for segmented regression. Indeed, it is possible to convert the algorithms of [GKS06] and [ADH$^+$15] to the segmented regression setting, but the corresponding guarantees are too weak to obtain good statistical results. At a high level, the algorithms in these papers can be adapted to output an $O(k)$-piecewise linear function $\widehat{f}$ so that $\sum_{i=1}^n (y_i - \widehat{f}(\boldsymbol{x}_i))^2 \le C \cdot \sum_{i=1}^n (y_i - f^{\mathrm{LS}}(\boldsymbol{x}_i))^2$, where $C > 1$ is a fixed constant and f^{LS} is the best k-piecewise linear least-squares fit to the data. However, this guarantee by itself does not give meaningful results for segmented regression.

As a toy example, consider the $k = 1$ case. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^n$ be arbitrary, let $f(\boldsymbol{x}) = 0$ for all $\boldsymbol{x}$, and $\epsilon_i \sim \mathcal{N}(0, 1)$, for $i = 1, \ldots, n$. Then it is not too hard to see that the least squares fit has squared error $\sum_{i=1}^n (y_i - f^{\mathrm{LS}}(\boldsymbol{x}_i))^2 = \Theta(n)$ with high probability. Hence the function g which is constantly $C/2$ for all $\boldsymbol{x}$ achieves the approximation guarantee

$$\sum_{i=1}^n (y_i - g(\boldsymbol{x}_i))^2 \;\le\; C \sum_{i=1}^n (y_i - f^{\mathrm{LS}}(\boldsymbol{x}_i))^2$$

described above with non-negligible probability. However, this function clearly does not converge to f as $n \to \infty$, and indeed its MSE is always $\Theta(1)$.

To get around this difficulty, we must extend the algorithms presented in histogram approximation literature in order to adapt them to the segmented regression setting. In the process, we introduce new algorithmic ideas and use more sophisticated proof techniques to obtain a meaningful rate of convergence for our algorithms.

2.3 Agnostic guarantees

We also consider the agnostic model, also known as learning with *model misspecification*. So far, we assumed that the ground truth is exactly a piecewise linear function. In reality, such a notion

is probably only an approximation. While the ground truth may be close to a piecewise linear function, generally we do not believe that it exactly follows a piecewise linear function. In this case, our goal should be to recover a piecewise linear function that is competitive with the best piecewise linear approximation to the ground truth.

Formally, we consider the following problem. We assume the same generative model as in (1), however, we no longer assume that f is a piecewise linear function. Instead, the function f can now be arbitrary. We define

$$\mathrm{OPT}_k = \min_{g \in \mathcal{L}_k} \mathrm{MSE}(g)$$

to be the error of the best fit k-piecewise linear function to f, and we let f^* be any k-piecewise linear function that achieves this minimum. Then the goal of our algorithm is to achieve an MSE as close to OPT_k as possible. We remark that the qualitatively interesting regime is when OPT_k is small and comparable to the statistical error, and we encourage readers to think of OPT_k as being in that regime.

For clarity of exposition, we first prove non-agnostic guarantees. In Section 6, we then show how to modify these proofs to obtain agnostic guarantees with roughly the same rate of convergence.

2.4 Mathematical Preliminaries

In this section, we state some preliminaries that our analysis builds on.

Tail bounds. We require the following tail bound on sub-exponential random variables. Recall that for any sub-exponential random variable X, the *sub-exponential norm* of X, denoted $\|X\|_{\psi_1}$, is defined to be the smallest parameter K so that $(\mathbb{E}[|X|^p])^{1/p} \leq Kp$ for all $p \geq 1$ (see [Ver10]). Moreover, if Y is a sub-Gaussian random variable with variance σ^2, then $Y^2 - \sigma^2$ is a centered sub-exponential random with $\|Y^2 - \sigma^2\|_{\psi_1} = O(\sigma^2)$.

Fact 6 (Bernstein-type inequality, c.f. Proposition 5.16 in [Ver10])**.** *Let $X_1, \ldots, X_N$ be centered sub-exponential random variables, and let $K = \max_i \|X_i\|_{\psi_1}$. Then for all $t > 0$, we have*

$$\Pr\left[\left|\frac{1}{\sqrt{N}}\sum_{i=1}^{N} X_i\right| \geq t\right] \leq 2\exp\left(-c\min\left(\frac{t^2}{K^2}, \frac{t\sqrt{N}}{K}\right)\right) .$$

This yields the following straightforward corollary:

Corollary 7. *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:*

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

Proof. Let $\delta' = O(\delta/n^2)$. For any interval $I \subseteq [n]$, by Fact 6 applied to the random variables $\epsilon_i^2 - s^2$ for $i \in I$, we have that (2) holds with probability $1 - \delta'$. Thus, by a union bound over all $O(n^2)$ subintervals of $[n]$, we conclude that Equation 2 holds with probability $1 - \delta$. □

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 8. *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 prediction error rate is known for the least-squares estimator:

Fact 9. *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 9 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 10 (c.f. proof of Theorem 2.2 in [Rig15])**.** *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:

Corollary 11. *Fix $\delta > 0$ and $\boldsymbol{v} \in \mathbb{R}^n$. Let $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ be as in Lemma 10. Then with probability at least $1 - \delta$, we have that for all intervals I, and for all non-zero linear functions $f : \mathbb{R}^d \to \mathbb{R}$,*

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

Proof. Fix any interval $I \subseteq [n]$. Observe that any linear function on I can only take values in the range $\{\boldsymbol{X}^I\boldsymbol{\theta} : \boldsymbol{\theta} \in \mathbb{R}^d\}$, and hence the range of functions of the form $f(\boldsymbol{x}_i) + \boldsymbol{v}$ is at most an r-dimensional affine subspace. Thus, by Lemma 10, we know that for any linear function f,

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

with probability $1 - O(\delta/n^2)$. By union bounding over all $O(n^2)$ intervals, we achieve the desired result. □

Corollary 12. *Fix $\delta > 0$ and $\boldsymbol{v} \in \mathbb{R}^n$. Let $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ be as in Lemma 10. Then with probability at least $1 - \delta$, we have*

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

Proof. Fix any partition of $[n]$ into k intervals $\mathcal{I}$, and let $S_{\mathcal{I}}$ be the set of k-piecewise linear functions which are flat on each $I \in \mathcal{I}$. It is not hard to see that $S_{\mathcal{I}}$ is a kr-dimensional linear subspace, and hence the set of all k-piecewise linear functions which are flat on each $I \in \mathcal{I}$ when translated by $\boldsymbol{v}$ is a kr-dimensional affine subspace. Hence Lemma 10 implies that

$$\sup_{f \in S_{\mathcal{I}}} \frac{|\langle \boldsymbol{\epsilon}, \boldsymbol{f}_I + \boldsymbol{v}_I \rangle|}{\|\boldsymbol{f}_I + \boldsymbol{v}_I\|} \;\leq\; O\left(\sigma\sqrt{kr + \log\frac{1}{\delta'}}\right)$$

with probability at least $1-\delta'$. A basic combinatorial argument shows that there are $\binom{n+k-1}{k-1} = n^{O(k)}$ such different partitions $\mathcal{I}$. Let $\delta' = \delta/\binom{n-k}{k}$. Then the result follows by union bounding over all the different possible partitions. □

2.5 Runtime of linear least squares

The appeal of linear least squares is not only its statistical properties, but also that the estimator can be computed efficiently. In our algorithm, we invoke linear least squares multiple times as a black-box subrountine. Primarily, we use the following theorem:

Fact 13. *Let $\boldsymbol{A} \in \mathbb{R}^{n \times d}$ be an arbitrary data matrix, and let $\boldsymbol{y}$ be the set of responses. Then there is an algorithm* LeastSquares(A, y) *that runs in time $O(nd^2)$ and computes the least squares fit to this data.*

There has been work on faster, approximate algorithms that would suffice for our purposes in theory. These algorithms offer a better dependence on the dimension d in exchange for slightly more complicated approximation guarantees and somewhat more complicated algorithms (for instance, see [CW13]). However, the classical algorithms for least squares with time complexity $O(nd^2)$ are more commonly used in practice. For this reason and to simplify our exposition, we thus present our results using the running time of the classical algorithms for least squares regression. Specifically, we write LeastSquares$(\boldsymbol{X}, \boldsymbol{y}, I)$ to denote an algorithm that computes a least squares fit on a given interval $I \subseteq [n]$ and assume that LeastSquares$(\boldsymbol{X}, \boldsymbol{y}, I)$ runs in time $O(|I| \cdot d^2)$ for all $I \subseteq [n]$.

3 Finding the least squares estimator via dynamic programming

In this section, we first present a dynamic programming approach (DP) to piecewise linear regression. We do not believe these results to be novel, but to the best of our knowledge, these results appear primarily as folklore in the literature. For completeness, we demonstrate the fastest DP we are aware of, and we also prove its statistical guarantees. Not only will this serve as a good warm-up for the later, more complex proofs, but we will also need a variant of this result in the later analyses.

3.1 The exact DP

We first describe the exact dynamic program. It will simply find the k-piecewise linear function which minimizes the sum-squared error to the data. In other words, it outputs

$$\arg\min_{f \in \mathcal{L}_k} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2 \quad = \quad \arg\min_{f \in \mathcal{L}_k} \|\boldsymbol{y} - \boldsymbol{f}\|^2 \ ,$$

which is simply the least-squares fit to the data amongst all k-piecewise linear functions. The dynamic program computes the estimator f as follows. For $i = 1, \ldots, n$ and $j = 1, \ldots, k$, let $A(i, j)$ denote the best error achievable by a j-piecewise linear function on the data pairs $\{(\boldsymbol{x}_\ell, y_\ell)\}_{\ell=1}^{i}$, that is, the best error achievable by j pieces for the first i data points. Then it is not hard to see that

$$A(i, j) = \min_{i' < i} \left(\mathrm{err}(\boldsymbol{X}, \boldsymbol{y}, \{i' + 1, \ldots, i\}) + A(i', j - 1) \right) \ ,$$

where for any interval I, we let $\mathrm{err}(\boldsymbol{X}, \boldsymbol{y}, I)$ denote the sum-squared error to the data of the best least squares fit to the data points $\{(\boldsymbol{x}_\ell, y_\ell)\}_{\ell \in I}$. That is, if f_I^{LS} is the least squares fit to the data in I, then $\mathrm{err}(\boldsymbol{X}, \boldsymbol{y}, I) = \|\boldsymbol{y}_I - \boldsymbol{f}_I^{\mathrm{LS}}\|^2$.

The algorithm then uses dynamic programming to fill out this $n \times k$ sized table of A values, starting at $i = 1$ and $j = 1$. After having done so, the algorithm does one additional pass backwards over the table to actually find the path through the table which achieves the best error. We can optimize this by first computing the error quantities $\mathrm{err}(\boldsymbol{X}, \boldsymbol{y}, \{i' + 1, \ldots, i\})$ for all $i' < i$, and then using a lookup table to find their values while actually executing the DP. Given such a lookup table, the DP runs in time $O(n^2 k)$. Naively, the construction of this look-up table would take time which is $O(n^3 d^2)$ since there are $O(n^2)$ linear regression problems of size $O(n)$, each of which takes $O(nd^2)$ time to solve.

However, we can speed this up. Consider a fixed interval $I \subset [n]$. Then the least squares fit on this interval is of the form $\boldsymbol{X}_I^T \boldsymbol{X}_I \boldsymbol{\theta}_I = \boldsymbol{X}_I^T \boldsymbol{y}_I$. Assuming the matrix $\boldsymbol{X}_I^T \boldsymbol{X}_I$ is invertible, we have that $\boldsymbol{\theta}_I = \left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1} \boldsymbol{X}_I^T \boldsymbol{y}_I$. Moreover, the error of the fit is given by

$$\begin{aligned} \|\boldsymbol{X}_I \boldsymbol{\theta}_I - \boldsymbol{y}_I\|^2 &= \left(\boldsymbol{X}_I \boldsymbol{\theta}_I - \boldsymbol{y}_I\right)^T \left(\boldsymbol{X}_I \boldsymbol{\theta}_I - \boldsymbol{y}_I\right) \\ &= \boldsymbol{\theta}_I^T \boldsymbol{X}_I^T \boldsymbol{X}_I \boldsymbol{\theta}_I - 2 \boldsymbol{y}^T \boldsymbol{X}_I \boldsymbol{\theta}_I + \boldsymbol{y}_I^T \boldsymbol{y}_I \\ &= \boldsymbol{y}_I^T \boldsymbol{X}_I \left[\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)\right]^{-1} \boldsymbol{X}_I^T \boldsymbol{y}_I - 2 \boldsymbol{y}_I^T \boldsymbol{X}_I \boldsymbol{\theta}_I + \boldsymbol{y}_I^T \boldsymbol{y}_I \ . \end{aligned}$$

The main point is the following: given $\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1}$ and $\boldsymbol{X}_I^T \boldsymbol{y}_I$, we can compute all the remaining quantities in time which is $O(d^2)$. To compute $\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1}$ and $\boldsymbol{X}_I^T \boldsymbol{y}_I$, we use the fact that our regression instances are not arbitrary but very closely related. As a result, we can re-use computation

from previous regression instances that we have already solved in order to speed up later regression instances.

Concretely, the calculations above indicate that we need to compute $\boldsymbol{X}_I^T \boldsymbol{y}_I$ for all intervals $I \subseteq [n]$. But these are all just sub-vectors of the vector $\boldsymbol{X}^T \boldsymbol{y}$, which we compute once in time $O(nd)$ and then use later on in all of the remaining computations. More non-trivially, we also need to compute $(\boldsymbol{X}_I \boldsymbol{X}_I)^{-1}$ for all $I \subseteq [n]$. However, observe that if we let $I = \{\ell, \ldots, p-1\}$ for $\ell < p$, then $\boldsymbol{X}_{I \cup \{p\}}^T \boldsymbol{X}_{I \cup \{p\}} = \boldsymbol{X}_I^T \boldsymbol{X}_I + \boldsymbol{x}_p \boldsymbol{x}_p^T$, so that adding a single data point to the data matrix corresponds to a rank-one update of the data matrix. Moreover, the effect of a rank-one update to the inverse of the matrix is well-known:

Theorem 14 (Sherman-Morrison formula)**.** *Suppose $\boldsymbol{M} \in \mathbb{R}^{d \times d}$ is an invertible square matrix, and let $\boldsymbol{v} \in \mathbb{R}^d$ be such that $1 + \boldsymbol{v}^T \boldsymbol{M}^{-1} \boldsymbol{v} \neq 0$. Then*

$$\left(\boldsymbol{M} + \boldsymbol{v}\boldsymbol{v}^T\right)^{-1} = \boldsymbol{M}^{-1} - \frac{\boldsymbol{M}^{-1}\boldsymbol{v}\boldsymbol{v}^T\boldsymbol{M}^{-1}}{1 + \boldsymbol{v}^T\boldsymbol{M}^{-1}\boldsymbol{v}} \; .$$

Thus, given $\boldsymbol{M}^{-1}$ and $\boldsymbol{v}$ satisfying these conditions, we may compute $\left(\boldsymbol{M} + \boldsymbol{v}\boldsymbol{v}^T\right)^{-1}$ in time $O(d^2)$.

Therefore, the dynamic program can do as follows: for each $\ell = 1, \ldots, n$, first compute $\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1}$ for the interval I of length d starting at ℓ, and then use the Sherman-Morrison update formula[2] to compute $\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1}$ for every interval I starting at ℓ in total time $O(nd^2)$. Thus the algorithm takes $O(n^2 d^2)$ time total to compute all of the $\left(\boldsymbol{X}_I^T \boldsymbol{X}_I\right)^{-1}$. As demonstrated earlier, this implies the following:

Theorem 15. *The exact dynamic program runs in time $O(n^2(d^2 + k))$.*

3.2 Error analysis for the exact DP

We now turn our attention to the learning rate of the exact DP. We show:

Theorem 16. *Let $\delta > 0$, and let f^{LS} be the k-piecewise linear estimator returned by the exact DP. Then, with probability $1 - \delta$, we we have that*

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

Proof. We follow the proof technique for convergence of linear least squares as presented in [Rig15]. Recall f is the true k-piecewise linear function. Then, by the definition of the least squares fit, we have that

$$\|\boldsymbol{y} - \boldsymbol{f}^{\mathrm{LS}}\|^2 \;\leq\; \|\boldsymbol{y} - \boldsymbol{f}\|^2 \;=\; \|\boldsymbol{\epsilon}\|^2 \; .$$

By expanding out $\|\boldsymbol{y} - \boldsymbol{f}^{\mathrm{LS}}\|^2$, we have that

$$\begin{aligned} \|\boldsymbol{y} - \boldsymbol{f}^{\mathrm{LS}}\|^2 \;&=\; \|\boldsymbol{f} + \boldsymbol{\epsilon} - \boldsymbol{f}^{\mathrm{LS}}\|^2 \\ &=\; \|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\|^2 + 2\left\langle \boldsymbol{\epsilon}, \boldsymbol{f} - \boldsymbol{f}^{\mathrm{LS}} \right\rangle + \|\boldsymbol{\epsilon}\|^2 \; . \end{aligned} \tag{3}$$

[2] In general, our matrices may not be invertible, or the update vector may not satisfy the condition in the Sherman-Morrison formula, but in practice it seems these issues don't come up and thus we do not worry about them here. In general there are more complicated formulas for rank one updates for pseudo-inverses here but we will not cover them for simplicity.

From these two calculations, we gather that

$$\begin{aligned}\|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\|^2 &\le 2\langle \boldsymbol{\epsilon}, \boldsymbol{f} - \boldsymbol{f}^{\mathrm{LS}}\rangle \\ &\le O\left(\sigma\sqrt{kr + k\log\frac{n}{\delta}}\right) \cdot \|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\| ,\end{aligned}$$

with probability $1-\delta$, where the last line follows from Corollary 12. A simple algebraic manipulation from this last inequality yields the desired statement. □

The same proof technique can also be easily adapted to yield the following slight extension of Theorem 16, which can be proven via a union bound over all sets of k disjoint intervals. We omit a proof here for conciseness.

Lemma 17. *Fix $\delta > 0$. Then with probability $1-\delta$ we have the following: for all disjoint sets of k intervals $I_1, \dots, I_k$ of $[n]$ so that f is flat on each I_ℓ, the following inequality holds:*

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

4 A simple greedy merging algorithm

In this section, we give a novel greedy algorithm which runs much faster than the DP, but which achieves a somewhat worse learning rate. However, we show both theoretically and experimentally that the tradeoff between speed and statistical accuracy for this algorithm is markedly better than it is for the exact DP.

4.1 The greedy merging algorithm

The overall structure of the algorithm is quite similar to [ADH+15], however, the merging criterion is different, and as explained above, the guarantees proved in that paper are insufficient to give non-trivial learning guarantees for regression.

Our algorithm here also requires an additional input s^2, which is defined to be the variance of the ϵ_i variables, i.e., $s^2 = \mathbb{E}[\epsilon_i^2]$. Requiring that we know s^2 is a drawback, and in Section 5 we give a slightly more complicated algorithm which does not require knowledge of s.

We give the formal pseudocode for the procedure in Algorithm 1. In the pseudocode we provide two additional tuning parameters τ, γ. This is because in general our algorithm cannot provide a k-histogram, but instead provides an $O(k)$ histogram, which for most practical applications suffices. The tuning parameters allow us to trade off running time for fewer pieces. In the typical use case we will have $\tau, \gamma = \Theta(1)$, in which case our algorithm will output an $O(k)$-piecewise linear function in time $O(nd^2 \log n)$ time.

4.2 Runtime of GreedyMerging

In this section we prove that our algorithm has the following, nearly-linear running time. The analysis is similar to the analysis presented in [ADH+15].

Theorem 18. *Let $\boldsymbol{X}$ and $\boldsymbol{y}$ be as in (1). Then* GreedyMerging$(\tau, \gamma, s, \boldsymbol{X}, \boldsymbol{y})$ *outputs a $\left((2+\frac{2}{\tau})k + \gamma\right)$-piecewise linear function and runs in time $O(nd^2 \log(n/\gamma))$.*

Algorithm 1 Piecewise linear regression by greedy merging.

function 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**
      $\boldsymbol{\theta}_u \leftarrow$ LEASTSQUARES($\boldsymbol{X}, \boldsymbol{y}, I_{2u-1} \cup I_{2u}$)
      $e_u = \|\boldsymbol{y}_I - \boldsymbol{X}_I \boldsymbol{\theta}_u\|_2^2 - s^2 |I_{2u-1} \cup I_{2u}|$
    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 $\mathcal{I}^j$
end function

Before we prove this theorem, we compare this with the running time for the exact DP as given in Theorem 15. Our main advantage is that our runtime scales linearly with n instead of quadratically. This manifests itself as a substantially win theoretically in most reasonable regimes, and also as a big win in practice—see our experiments for more details there.

Proof of Theorem 18. We first bound the time it takes to run any single iteration of the algorithm. In any iteration j, we do a linear least squares regression problem on each interval $I \in \mathcal{I}^j$; each such problem takes $O(|I|d^2)$ time; hence solving them all takes $O(nd^2)$ time. Computing the e_u given the least squares fit takes no additional time asymptotically, and finding the $\left(1 + \frac{1}{\tau}\right)$ largest errors takes linear time [CLRS09]. Afterwards the remaining computations in this iteration can easily be seen to be done in linear time. Hence each iteration takes $O(nd^2)$ time to complete.

We now bound the number of iterations of the algorithm. By the same analysis as that done in the proof of Theorem 3.4 in [ADH$^+$15] one can show that the algorithm terminates after at most $\log(n/\gamma)$ iterations. Thus the whole algorithm runs in time $O(nd^2 \log(n/\gamma))$ time, as claimed. □

4.3 Analysis of GreedyMerging

Theorem 19. *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 7, 11 and 12, and Lemma 17 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 17, we have

$$\sum_{I\in\mathcal{F}} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; O(\sigma^2|\mathcal{F}|(r + \log(n/\delta))) , \tag{4}$$

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 7 implies that the following inequality holds with probability at least $1 - O(\delta)$:

$$\begin{aligned}\sum_{I\in\mathcal{J}_1} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 &= \sum_{I\in\mathcal{J}_1} \|\epsilon_I\|^2 \\ &\le \sigma^2\left(\sum_{I\in\mathcal{J}_1} |I| + O\left(\log\frac{n}{\delta}\right)\sqrt{\sum_{I\in\mathcal{J}_1}|I|}\right) \\ &\le \sigma^2\left(m + O\left(\log\frac{n}{\delta}\right)\sqrt{m}\right) .\end{aligned} \tag{5}$$

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

$$\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| \;\le\; \|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\|^2 - s^2|M_\ell| . \tag{6}$$

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 $M_1, \ldots, M_{k/\tau}$.

Fix any $\ell = 1, \ldots, k/\tau$. Expanding out the RHS of (6) using the definition of y_i gives

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

Thus, we have that in aggregate,

$$\sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 - s^2|M_\ell| = \sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 + 2\sum_{\ell=1}^{k/\tau} \langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\rangle + \sum_{\ell=1}^{k/\tau} \sum_{i \in M_\ell} (\epsilon_i^2 - s^2)\,. \tag{7}$$

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 17 implies

$$\sum_{\ell=1}^{k/\tau} \left\|\boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 \le O\left(\sigma^2 \frac{k}{\tau}\left(r + \log\frac{n}{\delta}\right)\right)\,, \tag{8}$$

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 12 and (8) imply that

$$\begin{aligned}\sum_{\ell=1}^{k/\tau} \langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\rangle &\le \sqrt{\sum_{\ell=1}^{k/\tau} \langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{f}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\rangle} \cdot \sup_{\eta} \frac{|\langle \boldsymbol{\epsilon}_{M_\ell}, \boldsymbol{X}\eta\rangle|}{\|\boldsymbol{X}\eta\|} \\ &\le O\left(\sigma^2 \frac{k}{\tau}\left(r + \log\frac{n}{\delta}\right)\right)\,.\end{aligned} \tag{9}$$

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

By Corollary 7, 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

$$\sum_{i=1}^{k/\tau} \left(\left\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2 - s^2|M_\ell|\right) \le O\left(\frac{k}{\tau}\sigma^2\left(r + \log\frac{n}{\delta}\right)\right) + O\left(\sigma \log\frac{n}{\delta}\right)\sqrt{n} \tag{10}$$

with probability $1 - O(\delta)$. Since the LHS of (6) is bounded by each individual summand above, this implies that the LHS is also bounded by their average, which implies that

$$\begin{aligned}\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| \;&\le\; \frac{\tau}{k}\sum_{i=1}^{k/\tau} \left(\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\|^2 - s^2|M_\ell|\right) \\ &\le\; O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) + O\left(\sigma \log\frac{n}{\delta}\right)\frac{\tau\sqrt{n}}{k}\,.\end{aligned} \tag{11}$$

We now similarly expand out the LHS of (6):

$$\begin{aligned}\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| &= \|\boldsymbol{f}_I + \boldsymbol{\epsilon}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| \\ &= \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 + 2\langle \boldsymbol{\epsilon}_I, \boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\rangle + \|\boldsymbol{\epsilon}_I\|^2 - s^2|I| \,. \end{aligned} \tag{12}$$

From this we are interested in obtaining an upper bound on $\sum_{i\in I}(f(\boldsymbol{x}_i) - \widehat{f}(\boldsymbol{x}_i))^2$, hence we seek to lower bound the second and third terms of (12). The calculations here will very closely mirror those done above.

By Corollary 11, we have that

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

and by Corollary 7 we have

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

and so

$$\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\|^2 - s^2|I| \;\geq\; \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 - O\left(\sigma\sqrt{r + \log\left(\frac{n}{\delta}\right)}\right)\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\| - O\left(\sigma \log\frac{n}{\delta}\right)\sqrt{|I|} \,. \tag{13}$$

Putting (11) and (13) together yields that with probability $1 - O(\delta)$,

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

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

$$\begin{aligned} b &= O\left(\sigma\sqrt{r + \log\frac{n}{\delta}}\right) \,, \text{ and} \\ c &= O\left(\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right)\right) + O\left(\sigma\log\frac{n}{\delta}\right)\left(\frac{\tau\sqrt{n}}{k} + \sqrt{|I|}\right) \end{aligned}$$

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

Lemma 20. *Suppose $z^2 \leq bz + c$ where $b, c \geq 0$. Then $z^2 \leq O(b^2 + c)$.*

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

Thus, from the lemma, we have

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

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

$$\begin{aligned}\sum_{I\in\mathcal{J}_2}\|\boldsymbol{f}_I-\widehat{\boldsymbol{f}}_I\|^2 \;&\le\; O\left(k\sigma^2\left(r+\log\left(\frac{n}{\delta}\right)\right)\right)+O\left(\sigma\log\frac{n}{\delta}\right)\left(\tau\sqrt{n}+\sum_{I\in\mathcal{J}}\sqrt{|I|}\right)\\ &\le O\left(k\sigma^2\left(r+\log\left(\frac{n}{\delta'}\right)\right)\right)+O\left(\sigma\log\frac{n}{\delta}\right)\left(\tau\sqrt{n}+\sqrt{kn}\right)\;. \qquad (14)\end{aligned}$$

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 (4), (5), and (14) yields the desired conclusion. □

5 A variance-free merging algorithm

In this section, we give a variant of the above algorithm that does not require knowledge of the noise variance s^2. The formal pseudo code is given in Algorithm 2.

Algorithm 2 Variance-free greedy merging with bucketing.

function BucketGreedyMerging($\gamma, \boldsymbol{X}, \boldsymbol{y}$)
  ▷ Initial histogram.
  Let $\mathcal{I}^0 \leftarrow \{\{1\},\{2\},\ldots,\{n\}\}$ be the initial partition of $[n]$ into intervals of length 1.
  ▷ Iterative greedy merging (we start with $j=0$).
  while $|\mathcal{I}^j| > (2(k+1)+\gamma)\log n$ **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**
      $I'_u \leftarrow I_{2u-1}\cup I_{2u}$
      $\boldsymbol{\theta}_u \leftarrow$ LeastSquares$(\boldsymbol{X},\boldsymbol{y},I'_u)$
      $e_u = \frac{1}{|I'_u|}\|\boldsymbol{y}_I-\boldsymbol{X}_I\boldsymbol{\theta}_u\|_2^2$
    end for
    for $\alpha = 0,\ldots,\log(n)-1$ **do**
      Let B_α be the set of indices u so that $2^\alpha \le |I'_u| \le 2^{\alpha+1}$
      Let L_α be the set of indices $u\in B_\alpha$ with the $k+1$ largest e_u amongst $u\in B_\alpha$, breaking ties arbitrarily.
      Let M_α be the set of the remaining indices u in B_α.
      ▷ *Keep the intervals in each B_α with large merging errors.*
      $\mathcal{I}^{j+1} \leftarrow \bigcup_{u\in L_\alpha}\{I_{2u-1},I_{2u}\}$
      ▷ *Merge the remaining intervals in each B_α.*
      $\mathcal{I}^{j+1} \leftarrow \mathcal{I}^{j+1}\cup\{I'_u \,|\, u\in M_\alpha\}$
    end for
    $j \leftarrow j+1$
  end while
  return the least squares fit to the data on every interval in $\mathcal{I}^j$
end function

We now state the running time for BucketGreedyMerge. The running time analysis for BucketGreedyMerge is almost identical to that of GreedyMerge; hence we omit its proof.

Theorem 21. *Let $\boldsymbol{X}$ and $\boldsymbol{y}$ be as in* (1)*. Then* BucketGreedyMerging$(\gamma, \boldsymbol{X}, \boldsymbol{y})$ *outputs a $(2(k+1)+\gamma)\log n$-piecewise linear function and runs in time $O(nd^2\log(n/\gamma))$.*

5.1 Analysis of BucketGreedyMerge

This section is dedicated to the proof of the following theorem:

Theorem 22. *Let $\widehat{f}$ be the m-piecewise linear function that is returned by* BucketGreedyMerge*, where $m = (2(k+1)+\gamma)\log n$. Then, with probability $1-\delta$, we have*

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

Proof. As in the proof of Theorem 19, we let $\mathcal{I}$ be the final partition of $[n]$ that our algorithm produces. We also similarly condition on the event that Corollaries 7 and 12 and Lemma 17 all hold with parameter $O(\delta)$. We again partition $\mathcal{I}$ into two sets $\mathcal{F}$ and $\mathcal{J}$ as before, and further subdivide $\mathcal{J}$ into $\mathcal{J}_1$ and $\mathcal{J}_2$. The error on $\mathcal{F}$ and $\mathcal{J}_1$ is the same as the error given in (4) and (5). The only difference is the error on intervals in $\mathcal{J}_2$.

Let $I \in \mathcal{J}_2$ be fixed. By definition, this means there was some iteration and a collection of some $k+1$ disjoint intervals $M_1, \dots, M_{k+1}$ so that $|M_\ell|/2 \le |I| \le 2|M_\ell|$ and

$$\frac{1}{|I|}\left\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\right\|^2 \le \frac{1}{|M_\ell|}\left\|\boldsymbol{y}_{M_\ell} - \boldsymbol{f}^{\mathrm{LS}}_{M_\ell}\right\|^2$$

for all $\ell = 1, \dots, k+1$. Since f has at most k jumps, this means that there is at least one interval M_ℓ so that f is flat on M_ℓ. WLOG assume that f is flat on M_1. By the same kinds of calculations done in the proof of Theorem 19, we have that with probability $1-\delta$,

$$\left\|\boldsymbol{y}_{M_1} - \boldsymbol{f}^{\mathrm{LS}}_{M_1}\right\|^2 \le O\left(\sigma^2(r+\log\frac{n}{\delta})\right) + \sigma^2|M_1| + O(\sigma\log(n/\delta))\sqrt{|M_1|}$$

and

$$\left\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\right\|^2 \ge \left\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\right\|^2 - O\left(\sigma\sqrt{r+\log(n/\delta)}\right)\left\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\right\| + \sigma^2|I| - O(\sigma\log(n/\delta))\sqrt{|I|} ,$$

and putting these two equations together and rearranging, we have

$$\begin{aligned}\left\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\right\|^2 &\le O\left(\sigma^2(r+\log(n/\delta))\right) + \sigma^2\cdot O(\log(n/\delta))\left(\frac{|I|}{\sqrt{|M_\ell|}} + \sqrt{|I|}\right) \\ &\quad + O\left(\sigma\sqrt{r+\log(n/\delta)}\right)\left\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\right\| \\ &\le O\left(\sigma^2(r+\log(n/\delta))\right) + O\left(\sigma^2\log(n/\delta)\sqrt{|I|}\right) + O\left(\sigma\sqrt{r+\log(n/\delta)}\right)\left\|\boldsymbol{y}_I - \widehat{\boldsymbol{f}}_I\right\| ,\end{aligned}$$

where in the last inequality we used that $|M_\ell| \ge |I|/2$. By Lemma 20, this implies that

$$\left\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\right\|^2 \le O\left(\sigma^2(r+\log(n/\delta))\right) + O\left(\sigma^2\log(n/\delta)\sqrt{|I|}\right) .$$

Since there are at most k elements in $\mathcal{J}_2$, and they are all disjoint, this yields that

$$\sum_{I\in\mathcal{J}_2}\left\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\right\|^2 \le O\left(k\sigma^2\left(r+\log\frac{n}{\delta}\right)\right) + O\left(\sigma\log\left(\frac{n}{\delta}\right)\sqrt{kn}\right) \tag{15}$$

and putting this equation together with (4) and (5) yields the desired conclusion. □

5.2 Postprocessing

One unsatisfying aspect of BucketGreedyMerge is that it outputs $O(k \log n)$ pieces. Not only does this increase the size of the representation nontrivially when n is large, but it also increases the error rate: it is the reason why the first term in the error rate given in Theorem 22 has an additional $\log n$ factor as opposed the rate in Theorem 19. In this section, we give an efficient postprocessing procedure for BucketGreedyMerge which, when run on the output of BucketGreedyMerge, outputs an $O(k)$ histogram with the same rate as before. In fact, the rate is slightly improved as we are able to remove the $\log n$ factor mentioned above.

The postprocessing algorithm Postprocessing takes as input a partition $\mathcal{I}$ of $[n]$ of size $O(k \log n)$ and a target number of pieces k. It then performs the following steps: starting from the $O(k \log n)$ partition $\mathcal{I}$, run the dynamic program (DP) on intervals with breakpoints amongst the breakpoints of $\mathcal{I}$ to find the $2k+1$ partition $\mathcal{I}_p$ (whose endpoints are also endpoints of $\mathcal{I}$) that minimizes the sum squared error to the data. The running time analysis is identical to that of the DP with two exceptions: first, we only need to fill out a $O(k \log n) \times (2k+1)$ size table (as compared to an $n \times k$ sized table). Second, we are no longer performing rank one updates because we instead compute updates in large chunks, we cannot use the Sherman-Morrison formula to speed up the computation of the least-squares fits. Hence, we obtain the following theorem:

Theorem 23. *Given a partition $\mathcal{I}$ of $[n]$ into $O(k \log n)$ pieces,* Postprocessing$(\mathcal{I}, k)$ *runs in time $\widetilde{O}(k^3 d^2)$, and outputs a $(2k+1)$-piecewise linear function, where the $\widetilde{O}$ hides* $\operatorname{poly}\log(n)$ *factors.*

We now show that the algorithm still provides the same (in fact, slightly better) statistical guarantees as the original partition:

Theorem 24. *Fix $\delta > 0$. Let $\widehat{f}^p$ be the estimator output by* PostprocessedBucketGreedyMerge. *Then, with probability $1 - \delta$, we have*

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

Proof. Let $\mathcal{I}$ and $\mathcal{J}$ be as in the proof of Theorem 22. Let $\mathcal{I}_p = \{J_1, \ldots, J_{2k+1}\}$ be the intervals in the partition that we return. We again condition on the event that Corollaries 7 and 12 and Lemma 17 all hold with parameter $O(\delta)$. The following will then all hold with probability $1 - \delta$.

Define the partition $\mathcal{K}$ to be the partition that contains every interval in $\mathcal{J}$ and exactly one interval between any two non-consecutive intervals in $\mathcal{J}$ (i.e., $\mathcal{K}$ merges all flat intervals). Moreover, let g be the $(2k+1)$-piecewise linear function which is the least squares fit to the data on each interval in $I \in \mathcal{K}$. This is clearly a possible solution for the dynamic program, and therefore we have

$$\begin{aligned} \|\widehat{\boldsymbol{f}} - \boldsymbol{y}\|^2 &\leq \|\boldsymbol{g} - \boldsymbol{y}\|^2 \\ &= \sum_{I \in \mathcal{K} \setminus \mathcal{J}} \|\boldsymbol{g}_I - \boldsymbol{y}_I\|^2 + \sum_{I \in \mathcal{J}} \|\boldsymbol{g}_I - \boldsymbol{y}_I\|^2 . \end{aligned} \tag{16}$$

We will expand and then upper bound the RHS of (16). First, by calculations similar to those employed in the proof of Theorem 19, we have that

$$\sum_{I \in \mathcal{K} \setminus \mathcal{J}} \|\boldsymbol{g}_I - \boldsymbol{y}_I\|^2 \leq O\left(k\sigma^2\left(r + \log \frac{n}{\delta}\right)\right) + \sum_{I \in \mathcal{K} \setminus \mathcal{J}} \|\boldsymbol{\epsilon}_I\|^2 ,$$

and from (15) and Corollary 12 we have

$$\begin{aligned}\sum_{I\in\mathcal{J}}\|\boldsymbol{g}_I-\boldsymbol{y}_I\|^2 &= \sum_{I\in\mathcal{J}}\|\boldsymbol{f}_I+\boldsymbol{\epsilon}_I-\boldsymbol{g}_I\|^2\\ &= \sum_{I\in\mathcal{J}}\|\boldsymbol{f}_I-\boldsymbol{g}_I\|^2+2\sum_{I\in\mathcal{J}}\langle\boldsymbol{\epsilon}_I,\boldsymbol{f}_I-\boldsymbol{g}_I\rangle+\sum_{I\in\mathcal{J}}\|\boldsymbol{\epsilon}_I\|^2\\ &\le O\left(k\sigma^2\left(r+\log\frac{n}{\delta}\right)\right)+O\left(\sigma\log\left(\frac{n}{\delta}\right)\sqrt{kn}\right)+\sum_{I\in\mathcal{J}}\|\boldsymbol{\epsilon}_I\|^2\,,\end{aligned}$$

so all together now we have

$$\sum_{I\in\mathcal{K}}\|\boldsymbol{g}_I-\boldsymbol{y}_I\|^2\le O\left(k\sigma^2\left(r+\log\frac{n}{\delta}\right)\right)+O\left(\sigma\log\left(\frac{n}{\delta}\right)\sqrt{kn}\right)+\|\boldsymbol{\epsilon}\|^2\,.$$

Moreover, by the same kinds of calculations, we can expand out the LHS of (16):

$$\begin{aligned}\|\widehat{\boldsymbol{f}}-\boldsymbol{y}\|^2 &\ge \|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|^2+\langle\boldsymbol{\epsilon},\widehat{\boldsymbol{f}}-\boldsymbol{f}\rangle+\|\boldsymbol{\epsilon}\|^2\\ &\ge \|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|^2-O\left(\sqrt{k}\cdot\sigma\sqrt{r+\log\frac{n}{\delta}}\right)\|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|+\|\boldsymbol{\epsilon}\|^2\end{aligned}$$

and hence, combining, cancelling, and moving terms around, we get that

$$\|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|^2\le O\left(\sqrt{k}\cdot\sigma\sqrt{r+\log\frac{n}{\delta}}\right)\left(1+\|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|\right)+O\left(\sigma\log\left(\frac{n}{\delta}\right)\sqrt{kn}\right)\,.$$

By Lemma 20, this implies that

$$\|\widehat{\boldsymbol{f}}-\boldsymbol{f}\|^2\le O\left(k\sigma^2\left(r+\log\frac{n}{\delta}\right)+\sigma\log\left(\frac{n}{\delta}\right)\sqrt{kn}\right)\,,$$

with probability $1-\delta$, as claimed. □

We remark that PostprocessING can also be run on the output of GreedyMerging to decrease the number of pieces from $O(k)$ to $2k+1$ if so desired. The proof that it maintains similar statistical guarantees is almost identical to the one presented above.

6 Obtaining agnostic guarantees

In this section, we demonstrate how to show agnostic guarantees for the algorithms in the previous sections. Recall now f is arbitrary, and f^* is a k-piecewise linear function which obtains the best approximation in mean-squared error to f amongst all k-piecewise linear functions. For all $i=1,\dots,n$, define $\zeta_i=f(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)$ to be the error at data point i of the approximation, so that for all i, we have

$$y_i=f^*(\boldsymbol{x}_i)+\zeta_i+\epsilon_i\,. \tag{17}$$

By definition, we have that $\|\boldsymbol{\zeta}\|^2=n\cdot\mathrm{OPT}_k$.

As a warm-up, we first show the following agnostic guarantee for the exact DP:

Theorem 25. *Fix $\delta > 0$. Let f_{LS} be the k-piecewise linear function returned by the exact DP. Then, with probability $1-\delta$, we have*

$$\mathrm{MSE}(f_{\mathrm{LS}}) \le O\left(\sigma^2 \frac{kr + k\log\frac{n}{\delta}}{n} + \sigma\log\left(\frac{1}{\delta}\right)\sqrt{\frac{\mathrm{OPT}_k}{n}} + \mathrm{OPT}_k\right)$$

In the regimes that are most interesting, i.e., when OPT_k is small, the middle term does not contribute significantly to the error.

Proof. The overall proof structure stays the same. From the definition of the least-squares fit, we have

$$\begin{aligned}
\|\boldsymbol{y} - \boldsymbol{f}^{\mathrm{LS}}\|^2 &\le \|\boldsymbol{y} - \boldsymbol{f}^*\|^2 \\
&= \|\boldsymbol{\epsilon} + \boldsymbol{\zeta}\|^2 \\
&= \|\boldsymbol{\epsilon}\|^2 + 2\langle\boldsymbol{\epsilon}, \boldsymbol{\zeta}\rangle^2 + n\cdot\mathrm{OPT}_k \\
&\le \|\boldsymbol{\epsilon}\|^2 + O\left(\sigma\log\frac{1}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k\,,
\end{aligned}$$

with probability $1 - O(\delta)$, where the last inequality follows from the $r = 1$ case of Lemma 10. The LHS can be expanded out exactly as before in (3), and putting these two sides together we obtain that

$$\begin{aligned}
\|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\|^2 \;&\le\; 2\langle\boldsymbol{\epsilon}, \boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\rangle + O\left(\sigma\log\frac{1}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k \\
&\le O\left(\sigma\sqrt{kr + k\log\frac{n}{\delta}}\right)\cdot\|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\| + O\left(\sigma\log\frac{1}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k
\end{aligned}$$

with probability at least $1 - O(\delta)$ and so by Lemma 20 we obtain that

$$\|\boldsymbol{f}^{\mathrm{LS}} - \boldsymbol{f}\|^2 \le O\left(\sigma^2\left(kr + k\log\frac{n}{\delta}\right) + \sigma\log\left(\frac{1}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k\right)\,.$$

Dividing both sides by n yields the desired conclusion. □

Reviewing the proof, we observe the following general pattern: in all upper bounds we wish to obtain (generally for formulas which occur on the RHS of the equations above) we obtain new OPT_k terms, whereas in the lower bounds (generally for formulas on the LHS of the equations above) nothing changes. In fact, all folllowing proofs exhibit this pattern. We first establish some useful concentration bounds. By the same union-bound technique used throughout this paper, one can show the following. We omit the proof for conciseness.

Lemma 26. *Fix $\delta > 0$. Let ϵ_i and ζ_i be as in (17), for $i = 1, \ldots, n$. Then, with probability $1-\delta$, we have that for all collections of k disjoint intervals $J_1, \ldots, J_k$, the following holds:*

$$\begin{aligned}
\sum_{\ell=1}^{k}\langle\boldsymbol{\epsilon}_{J_\ell}, \boldsymbol{\zeta}_{J_\ell}\rangle &\le O\left(k\sigma\log\frac{n}{\delta}\right)\cdot\left(\sum_{\ell=1}^{k}\|\boldsymbol{\zeta}_{J_\ell}\|^2\right)^{1/2} \\
&\le O\left(k\sigma\log\frac{n}{\delta}\right)\cdot(n\cdot\mathrm{OPT}_k)^{1/2}
\end{aligned}$$

Observe that with this lemma, we can easily adapt the proof of Theorem 25 to show the following agnostic version of Lemma 17:

Lemma 27. *Fix $\delta > 0$. Then with probability $1-\delta$, we have that for all disjoint sets of k intervals $J_1, \dots, J_k$ of $[n]$ so that f^* is flat on each J_ℓ, the following inequality holds:*

$$\sum_{\ell=1}^{k} \|\boldsymbol{f}_{J_\ell}^{\mathrm{LS}} - \boldsymbol{f}_{J_\ell}\|^2 \le O\left(\sigma^2\left(kr + k\log\frac{n}{\delta}\right) + k\sigma\log\left(\frac{n}{\delta}\right)(n\cdot \mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k\right) .$$

With this lemma, we can now prove the following theorem for our greedy algorithm, which shows that in the presence of model misspecification, our algorithm still performs at a good rate:

Theorem 28. *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*

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

Proof. As before, we condition on the event that Corollaries 7, 11, and 12, and Lemmas 26 and 27 all hold with error parameter $O(\delta)$, so that together they all hold with probability at least $1-\delta$. We also let $\mathcal{I}, \mathcal{F}, \mathcal{J}_1$, and $\mathcal{J}_2$ denote the same quantities as before, except we define the partition with respect to the jumps of f^* instead of f, since the latter does not have well-defined jumps. For instance, $\mathcal{F}$ is the set of intervals in $\mathcal{I}$ on which f^* has no jumps.

We again bound the error on these three sets separately. First, by Lemma 27 we have that

$$\sum_{I\in\mathcal{F}} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \;\le\; O\left(\sigma^2\left(mr + m\log\frac{n}{\delta}\right) + m\sigma\log\left(\frac{n}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + n\cdot\mathrm{OPT}_k\right) . \tag{18}$$

Second, we bound the error on $\mathcal{J}_1$. By modifying the calculations as those preceding (5) in the same way as we did above in the proof of Theorem 25, we may show that

$$\sum_{I\in\mathcal{J}_1} \|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \le \sigma^2\left(k + O\left(\log\frac{n}{\delta}\right)\sqrt{m}\right) + O\left(k\sigma\log\frac{n}{\delta}(n\cdot\mathrm{OPT}_k)^{1/2}\right) + n\cdot\mathrm{OPT}_k . \tag{19}$$

Finally, we bound the error on $\mathcal{J}_2$. Fix an $I \in \mathcal{J}_2$, and let $M_1, \dots, M_{k/\tau}$ be as before. Recall that this proof had two components: an upper bound for the $M_1, \dots, M_\ell$ and a lower bound for I. We first compute the upper bound. By following the calculations for (11) and using Lemma 27, we have

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

The lower bound is in fact unchanged: we may use (13) as stated. Putting these two terms together and simplifying as we did previously, we obtain that

$$\begin{aligned}\|\boldsymbol{f}_I - \widehat{\boldsymbol{f}}_I\|^2 \le O\Bigg(&\sigma^2\left(r + \log\left(\frac{n}{\delta}\right)\right) + \sigma\log\frac{n}{\delta}\left(\frac{\tau\sqrt{n}}{k} + \sqrt{|I|}\right) \\ &+\sigma\log\left(\frac{n}{\delta}\right)(n\cdot\mathrm{OPT}_k)^{1/2} + \frac{\tau}{k}n\cdot\mathrm{OPT}_k\Bigg) .\end{aligned}$$

As before, summing up all the bounds we have achieved proves the desired claim. □

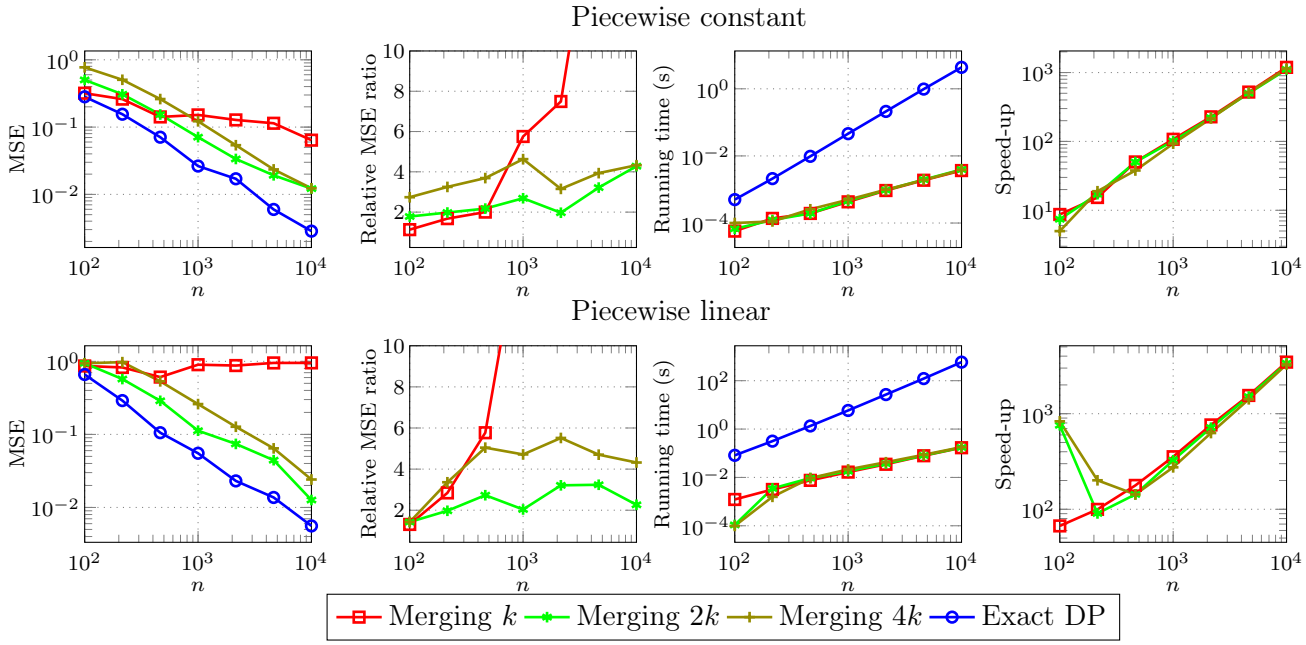

Figure 1: Experiments with synthetic data: results for piecewise constant models with $k = 10$ segments (top row) and piecewise linear models with $k = 5$ segments (bottom row, dimension $d = 10$). Compared to the exact dynamic program, the MSE achieved by the merging algorithm is worse but stays within 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).

Through virtually identical methods we also obtain the same upper bound for BucketGreedyMerge and PostprocessedBucketGreedyMerge. We state the results but omit their proofs for this reason.

Theorem 29. *Let $\widehat{f}$ be the m-piecewise linear function that is returned by* BucketGreedyMerge*, where $m = (2 + \frac{2}{\tau})k \log n + \gamma$. Then, with probability $1 - \delta$, we have*

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

Theorem 30. *Fix $\delta > 0$. Let $\widehat{f}_p$ be the estimator output by* PostprocessedBucketGreedyMerge*. Then, with probability $1 - \delta$, we have*

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

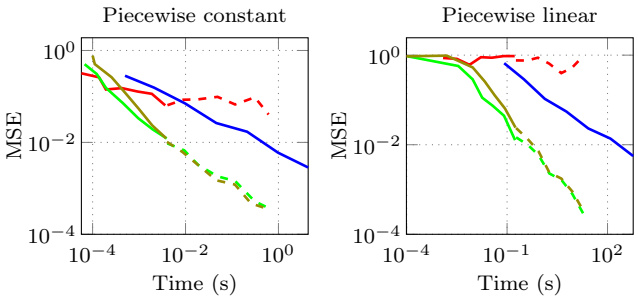

Figure 2: Computational vs. statistical efficiency in the synthetic data experiments. The solid lines correspond to the data in Figure 1, the dashed lines show the results from additional runs of the merging algorithms for larger values of n. The merging algorithm achieves the same MSE as the dynamic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

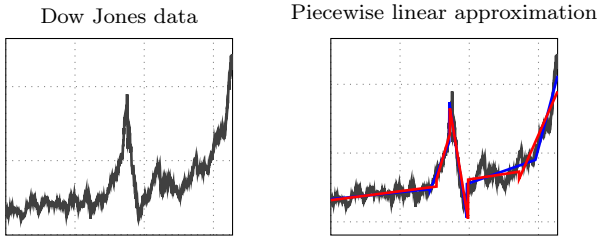

Figure 3: Results of fitting a 5-piecewise linear function ($d = 2$) to a Dow Jones time series. The merging algorithm produces a fit that is comparable to the dynamic program and is about 200× faster (0.013 vs. 3.2 seconds).

7 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[3] (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 piecewise linear function f with $d = 10$ features.

We generate the piecewise constant function f by randomly choosing $k = 10$ integers from the

[3] `http://julialang.org/`

set $\{1, \dots, 10\}$ as function value in each of the k segments.[4] 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).

Acknowledgements

Part of this research was conducted while Ilias Diakonikolas was at the University of Edinburgh, Jerry Li was an intern at Microsoft Research Cambridge (UK), and Ludwig Schmidt was visiting the EECS department 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.

[4] 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.

References

[ADH+15] J. Acharya, I. Diakonikolas, C. Hegde, J. Z. Li, and L. Schmidt. Fast and near-optimal algorithms for approximating distributions by histograms. In *PODS*, pages 249–263, 2015.

[ADLS15] J. Acharya, I. Diakonikolas, J. Zheng Li, and L Schmidt. Sample-optimal density estimation in nearly-linear time. *CoRR*, abs/1506.00671, 2015.

[ASW13] H. Avron, V. Sindhwani, and D. Woodruff. Sketching structured matrices for faster nonlinear regression. In *NIPS*, pages 2994–3002. 2013.

[BP98] J. Bai and P. Perron. Estimating and testing linear models with multiple structural changes. *Econometrica*, 66(1):47–78, 1998.

[CGS15] S. Chatterjee, A. Guntuboyina, and B. Sen. On risk bounds in isotonic and other shape restricted regression problems. *Annals of Statistics*, 43(4):1774–1800, 08 2015.

[CLRS09] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* 3rd edition, 2009.

[CW13] K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *STOC*, 2013.

[Fed75] P. I. Feder. On asymptotic distribution theory in segmented regression problems– identified case. *Annals of Statistics*, 3(1):49–83, 01 1975.

[Fri91] J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67, 03 1991.

[GA73] A. R. Gallant and Fuller W. A. Fitting segmented polynomial regression models whose join points have to be estimated. *Journal of the American Statistical Association*, 68(341):144–147, 1973.

[GKS06] S. Guha, N. Koudas, and K. Shim. Approximation and streaming algorithms for histogram construction problems. *ACM Trans. Database Syst.*, 2006.

[JKM+98] H. V. Jagadish, Nick Koudas, S. Muthukrishnan, Viswanath Poosala, Kenneth C. Sevcik, and Torsten Suel. Optimal histograms with quality guarantees. In *VLDB '98*, 1998.

[Jor13] M. I. Jordan. On statistics, computation and scalability. *Bernoulli*, 19(4):1378–1390, 09 2013.

[KRS15] R. Kyng, A. Rao, and S. Sachdeva. Fast, provable algorithms for isotonic regression in all l_p-norms. In *NIPS*, pages 2701–2709, 2015.

[Mey08] M. C. Meyer. Inference using shape-restricted regression splines. *Annals of Applied Statistics*, 2(3):1013–1033, 09 2008.

[MT77] F. Mosteller and J. W. Tukey. *Data analysis and regression: a second course in statistics.* Addison-Wesley, Reading (Mass.), Menlo Park (Calif.), London, 1977.

[Rig15] P. Rigollet. High dimensional statistics. 2015.

[SHKT97] C. J. Stone, M. H. Hansen, C. Kooperberg, and Y. K. Truong. Polynomial splines and their tensor products in extended linear modeling: 1994 wald memorial lecture. *Annals of Statistics*, 25(4):1371–1470, 1997.

[Sto94] C. J. Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *Annals of Statistics*, 22(1):pp. 118–171, 1994.

[Ver10] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. Chapter 5 of: Compressed Sensing, Theory and Applications. Edited by Y. Eldar and G. Kutyniok. Cambridge University Press, 2012, 2010.

[WW83] E. J. Wegman and I. W. Wright. Splines in statistics. *Journal of the American Statistical Association*, 78(382):pp. 351–365, 1983.

[YP13] Y. Yamamoto and P. Perron. Estimating and testing multiple structural changes in linear models using band spectral regressions. *Econometrics Journal*, 16(3):400–429, 2013.