Submitted to the Annals of Statistics

A CROSS VALIDATION FRAMEWORK FOR SIGNAL DENOISING WITH APPLICATIONS TO TREND FILTERING, DYADIC CART AND BEYOND

BY ANAMITRA CHAUDHURI* AND SABYASACHI CHATTERJEE†

*Department of Statistics, University of Illinois at Urbana Champaign, *ac34@illinois.edu; †sc1706@illinois.edu*

This paper formulates a general cross validation framework for signal denoising. The general framework is then applied to nonparametric regression methods such as Trend Filtering and Dyadic CART. The resulting cross validated versions are then shown to attain nearly the same rates of convergence as are known for the optimally tuned analogues. There did not exist any previous theoretical analyses of cross validated versions of Trend Filtering or Dyadic CART. To illustrate the generality of the framework we also propose and study cross validated versions of two fundamental estimators; lasso for high dimensional linear regression and singular value thresholding for matrix estimation. Our general framework is inspired by the ideas in [8] (Chatterjee and Jafarov, 2015) and is potentially applicable to a wide range of estimation methods which use tuning parameters.

1. Introduction. Cross Validation (CV) is a general statistical technique for choosing tuning parameters in a data driven way and is heavily used in practice for a wide variety of statistical methods. In spite of this, there is very little theoretical understanding of most CV algorithms used in practice. Within the nonparametric regression literature, rigorous theoretical guarantees for cross validated methods are limited to kernel smoothers, local linear regression methods or ridge regression (see [50], [26], [14]) which are all linear functions of the data y. There appears to be a need for theoretically backed general framework for building cross validation procedures for modern nonlinear regression methods. In this paper we attempt to start filling this gap in the literature by providing a general recipe to build provably adaptive and rate optimal CV estimators for some nonparametric estimation methods of current interest.

As an illustrative modern nonparametric regression method, we consider Trend Filtering (TF), proposed by [24]; see [43] for a comprehensive overview. TF estimators, of order $r \geq 1$, fit rth degree (discrete) splines (piecewise polynomials with certain regularity). In contrast to classical nonparametric regression methods such as local polynomials, splines, kernels etc., TF is a *spatially adaptive* method as the knots of the piecewise polynomials are chosen in a data driven fashion. The last few years have seen a flurry of research (e.g [42], [16], [31]) in trying to understand the theoretical properties of TF. However, all the existing guarantees hold when the tuning parameter is chosen in an optimal way depending on problem parameters which are typically unknown. On the other hand, the practical applications of TF almost always involves cross validating the tuning parameter. This motivates the following natural question. ***Is it possible to define a cross validated version of Trend Filtering which provably maintains all the risk guarantees known for optimally tuned Trend Filtering?*** This is an important open question which motivates the study in this paper.

Our main focus is on developing theoretically tractable CV versions for modern fixed design nonparametric regression/signal denoising methods such as Trend Filtering, Dyadic CART, other image/matrix denoising methods, etc. Inspired by the idea underlying the cross

MSC2020 subject classifications: Primary 62G05, 62G08.

Keywords and phrases: Cross Validation, Trend Filtering, Dyadic CART, Singular Value Thresholding, Lasso, Adaptive Risk Bounds.

validation method for Lasso, proposed by [8], we formalize a general cross validation framework for estimators in the so called *sequence model*. This framework, a variant of K fold CV, provides a unified, theoretically principled and computationally efficient way to design CV versions for a variety of estimation methods. In particular, we establish a general result about any CV estimator (which fits in our framework) in Theorem 2.1 which can then be used to obtain rate optimal guarantees for different estimators of interest.

We use this framework to propose and study a cross validated version of Trend Filtering with nearly matching theoretical guarantees known for the corresponding optimally tuned version; thereby answering our main question (in bold) posed above in the affirmative. To the best of our knowledge, before our work there has been no study done on the theoretical properties of a cross validated version of Trend Filtering. In practice, a particular CV version, implemented in the Rpackage Genlasso [1], is commonly used. However, no theoretical guarantees are available for this particular version. We outline the differences and similarities of our CV version with this one and present simulations which suggest that our CV version exhibits competitive finite sample performance as compared to this version.

We then use this framework to propose and study a cross validated version of Dyadic CART (DC), a classical regression tree method originally proposed in [12]. In a sense, DC can be thought of as an ℓ_0 penalized version of Trend Filtering which is an (generalized) ℓ_1 penalized least squares estimator. In [6], DC has been shown to be a computationally faster and statistically competitive alternative to Trend Filtering and its multivariate versions such as the Total Variation Denoising estimator (proposed by [37] and used heavily for image processing). This makes it natural for us to consider Dyadic CART alongside Trend Filtering in this paper. In spite of Dyadic CART being a classical nonparametric regression method and having been applied in various settings over the years; all the available theoretical results depend on a theoretical choice of the tuning parameter λ which depends on unknown problem parameters. We again show our cross validated version is able to attain nearly the same risk bound as is known for the optimally tuned one.

Trend Filtering and Dyadic CART are the two prime examples considered in this paper where we apply our general CV framework. However, our CV framework is quite general and is potentially applicable to any other method which uses tuning parameters. To illustrate the generality and flexibility of our CV framework, we further consider two fundamental estimation methods, Lasso for high dimensional regression and Singular Value Thresholding for Matrix Estimation. We propose and study new cross validated versions of these fundamental methods. In the case of the Lasso, our cross validated version can be thought of as the penalized counterpart of the estimator proposed in [8] which cross validates constrained Lasso. We show our cross validated Lasso estimator enjoys both types of standard rates of convergence known for optimally tuned Lasso. We finally consider matrix estimation by Singular Value Thresholding which is a canonical matrix estimation method; see [3], [11], [5]. We use our cross validation framework to derive a cross validated version of Singular Value Thresholding and provide rigorous adaptivity guarantees for it.

To summarize, this paper gives a general framework for cross validation and presents one general risk bound (Theorem 2.1) for any CV version of an estimation method which is built within our framework. Then we consider four different estimation methods, namely a) Trend Filtering, b) Dyadic CART, c) Lasso and d) Singular Value Thresholding. For each of these methods, we show how to construct a CV version within our framework. Next, we show how to apply Theorem 2.1 to our CV versions and establish rate optimality and adaptivity which is only known for the optimally tuned analogues of these methods. Essentially, our results for these estimators look like the one below (stated informally),

THEOREM 1.1 (Informal). *Let $\hat{\theta}$ be an optimally tuned estimation method (any one of the four stated above). Let $\hat{\theta}_{CV}$ be our CV version. Then, with high probability,*

$$MSE(\hat{\theta}_{CV}, \theta^*) \leq MSE(\hat{\theta}, \theta^*)\, p(\log n)$$

where θ^* denotes the true signal, MSE denotes the usual mean squared error and $p(\log n)$ is a (low degree) polynomial factor of $\log n$ where n is the sample size.

Outline: This paper is organized as follows. In Section 2 we describe and explain our cross validation framework in detail. We also give a general risk bound (see Theorem 2.1) for any CV estimator which falls under the scope of our framework in this section. We also provide a sketch of proof of Theorem 2.1 in this section. In Section 3 we propose a CV version of Dyadic CART and establish an oracle risk bound for it which is only known for an optimally tuned Dyadic CART. One of the attractive aspects of Dyadic CART is fast computation and in this section we similarly establish fast computation for our CV version by providing an algorithm in Section B in the supplementary file. In Section 4 we propose a CV version of Trend Filtering and establish both the so-called slow and fast rates known for optimally tuned Trend Filtering. In Sections 5 and 6 we propose CV versions of Singular Value Thresholding (SVT) and Lasso, and establish that they enjoy similar theoretical guarantees as are known for the optimally tuned versions. Section 7 discusses some matters naturally related to the research in this article. Section 8 contains simulations done for the CV versions of Dyadic CART and Trend Filtering proposed here.

The proofs of all our results are contained in the supplementary file. For instance, Section A contains the proof of our general risk bound (which is Theorem 2.1). Sections C, D, E and F contain the proofs of the risk bounds shown for Dyadic CART, Trend Filtering, SVT and Lasso respectively. For the convenience of the reader, we have provided proof sketches at the beginning of Sections C and D.

Notation: Throughout the paper we use the usual $O(\cdot)$ notation to compare sequences. We write $a_n = O(b_n)$ if there exists a constant $C > 0$ such that $a_n \leq C b_n$ for all sufficiently large n. We also use $a_n = \tilde{O}(b_n)$ to denote $a_n = O(b_n (\log n)^C)$ for some $C > 0$. The O_d notation is the same as the O notation except it signifies that the constant factor while comparing two sequences in n may depend on the underlying dimension d. For an event A, we will denote $1(A)$ to denote the indicator random variable of the event A.

We use C to denote a universal constant throughout the paper. This will be a positive constant independent of the problem parameters unless otherwise stated. The precise value of the constant C may change from line to line. We use $[m]$ denote the set of positive integers from 1 to m. For any vector $v \in \mathbb{R}^m$, we denote its ℓ_2 norm to be $\|v\| = \sqrt{\sum_{i=1}^m v_i^2}$. Similarly, we use $\|v\|_0, \|v\|_1$ and $|v|_\infty$ to denote its ℓ_0, ℓ_1 and ℓ_∞ norms respectively. Also, for any subset $S \subset [m]$, we use v_S to denote the vector in $\mathbb{R}^{|S|}$ obtained by restricting v to the coordinates in S. For any two vectors $v, v' \in \mathbb{R}^m$ we denote $\|v - v'\|^2$ by $SSE(v, v')$ where SSE stands for sum of squared errors. We denote the set of all positive real numbers by $\mathbb{R}_+$.

2. Cross Validation Framework. The precise setting we consider is that of signal denoising or fixed design regression where we observe $y = \theta^* + \epsilon$, where all these are $n \times 1$ vectors or vectorized matrices/tensors. θ^* is the true signal and ϵ is the noise vector consisting of i.i.d mean 0 subgaussian noise with subgaussian norm σ. This model is sometimes called the *subgaussian sequence model* and we will use the notation $y \sim Subg(\theta^*, \sigma^2)$ to mean that y arises from this probabilistic model. The precise distribution of the *errors* ϵ could be anything as long as subgaussianity is satisfied. The problem is to denoise or estimate the signal θ^* after observing y. Many well known and popular methods to estimate θ^* in this model involve the use of tuning parameters. For such methods, we now lay out our general K fold cross validation framework.

2.1. ***A General Framework of Cross Validation***. The following 6 general steps constitute our CV framework. This is a variant of $K \geq 2$ fold CV and is different from the typical K fold CV in some respects. Let $\hat{\theta}^{(\lambda)}$ be a given family of estimators (with tuning parameter λ) for which a CV version is desired.

1. Choose the number of folds K.
2. Partition $[n]$ into K disjoint index sets or folds $I_1, I_2, \ldots, I_K$. We allow this division to be done in a deterministic way or by using additional randomization. For any $j \in [K]$, denote I_j^c to be the index set which excludes the indices in I_j, that is $I_j^c = [n] \setminus I_j$.
3. For each $j \in [K]$ and any choice of a tuning parameter $\lambda_j > 0$, construct a version of $\hat{\theta}^{(\lambda_j)}$ which only depends on the data y through the coordinates in I_j^c or in other words is a function of $y_{I_j^c}$. We denote this estimator by $\hat{\theta}^{(\lambda_j, I_j^c)} \in \mathbb{R}^n$.
4. For each $j \in [K]$, choose a finite set of possible candidate values of the tuning parameter λ_j, namely Λ_j. The set Λ_j can be chosen deterministically or even in a data driven way as a function of $y_{I_j^c}$.
5. For any $j \in [K]$, denote the total squared prediction error (as a function of λ) on the jth fold by

$$CVERR_j(\lambda) = \left|\left| y_{I_j} - \hat{\theta}^{(\lambda, I_j^c)}_{I_j} \right|\right|^2.$$

   Define $\hat{\lambda}_j$ to be the candidate in Λ_j for which the prediction error on the jth fold is the minimum, that is,

$$\hat{\lambda}_j := \operatorname*{arg\,min}_{\lambda \in \Lambda_j} CVERR_j(\lambda).$$

   Now define an intermediate estimator $\tilde{\theta} \in \mathbb{R}^n$ such that

$$\tilde{\theta}_{I_j} = \hat{\theta}^{(\hat{\lambda}_j, I_j^c)}_{I_j}, \quad j \in [K]. \tag{1}$$

6. Define $\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \|\hat{\theta}^{(\lambda)} - \tilde{\theta}\|^2$ where $\Lambda \subseteq \mathbb{R}_+$ is a deterministic set of candidate tuning parameter values to be chosen by the user. Now define the final estimator $\hat{\theta}_{CV} \in \mathbb{R}^n$ to be

$$\hat{\theta}_{CV} = \hat{\theta}^{(\hat{\lambda})}. \tag{2}$$

We now discuss about various aspects of our K fold CV scheme and how it differs from the typical K fold CV scheme.

- Dividing the dataset into folds is typically done randomly which is natural when the design is random. Since our focus is on fixed design methods or signal denoising methods our framework is a bit more general and allows deterministic folds as well. In our application to Trend Filtering we prefer using a simple deterministic strategy to create the folds. This avoids the use of extra randomization and makes our estimator conceptually simpler. In other applications such as Low Rank Matrix Estimation and Lasso it is not clear if there is a sensible way to create folds deterministically and thus we propose to create the folds randomly.
- In any cross validation procedure, one needs to predict on a part of the data based on observations from the rest of the data. In our framework, the way one does this is by constructing estimators which are a function of a strict subset of the data. In particular, for each fold $j \in [K]$, the user needs to define estimators $\hat{\theta}^{(\lambda, I_j^c)} \in \mathbb{R}^n$ of θ^* which are functions only of $y_{I_j^c}$. Estimating the true signal θ^* based on only a subset of the data y can be thought of as a *completion* problem. Thus, we refer to the estimators $\hat{\theta}^{(\lambda, I_j^c)}$ as

completion estimators. One can use these completion estimators to define the predictions on the jth fold as $\hat{\theta}_{I_j}^{(\lambda,I_j^c)}$. As an example, if the estimation method under consideration is Trend Filtering, then the user needs to design completion versions of Trend Filtering which are based only on a strict subset of the data. How exactly can one define these completion versions is problem specific and is described later.

- In our framework, for each fold $j \in [K]$, the user needs to build a finite set of candidate tuning values Λ_j which is allowed to depend on $y_{I_j^c}$. This gives quite a bit of flexibility to the user. For example, for Trend Filtering we can use a particular data driven choice of Λ_j (see the discussion in Section 7.3.1). However, it should be said here that for all of our estimators, we find that setting $\Lambda_j = \Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ with $N^* = O(\log n)$, a simple deterministic exponentially growing grid, is sufficient for our purposes.
- In traditional K fold CV, a single optimized tuning parameter is commonly chosen by taking $\Lambda_j = \Lambda$ and by minimizing the sum of prediction errors over all the folds, that is,

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \sum_{j=1}^{K} CVERR_j(\lambda).$$

In contrast, in our framework we first construct K optimized tuning parameters $\hat{\lambda}_j$ (one for each fold) which minimize the prediction error on each fold. We then construct an intermediate interleaved estimator $\tilde{\theta}$ by gluing together the optimized fits on each of the folds as in (1). Finally, we then come up with a single optimized tuning parameter by minimizing the squared distance of $\hat{\theta}^{(\lambda)}$ (over a set Λ) to the intermediate fit $\tilde{\theta}$ as in (2). This seemingly roundabout way of choosing $\hat{\lambda}$ makes our cross validation scheme theoretically tractable; see our explanation in Section 2.3.

- The main advantage of our variant of K fold CV versus the traditional version of K fold CV is mathematical tractability (see Theorem 2.1 below). We believe that ingredients of the theoretical analysis of our variant could be a stepping stone towards a theoretical analysis of other CV versions used in practice. Furthermore, our simulations suggest that our CV versions not only enjoy rigorous theoretical guarantees but are also practically useful, providing good finite sample performance. For example, we found that in our simulations for Trend Filtering (see Section 8), the practical performance of our CV variant is very similar with the state of the art CV version implemented by the R package [1].

2.2. ***A General Result***. We now describe the main theoretical result underlying our cross validation framework. This result is our main tool and is used throughout the paper. This result bounds the squared error loss of the cross validated estimator $\hat{\theta}_{CV}$ defined in (2).

THEOREM 2.1. *Let $\hat{\theta}^{(\lambda)}$ be a given family of estimators in the subgaussian sequence model for a tuning parameter λ ranging in the set $\mathbb{R}_+$. Then the K fold cross validated estimator $\hat{\theta}_{CV}$ defined in (2) satisfies for all $x \geq 0$, with probability not less than $1 - 2K\exp(-x^2/2)$, the following inequality:*

$$\|\hat{\theta}_{CV} - \theta^*\| \leq \min_{\lambda \in \Lambda} \|\hat{\theta}^{(\lambda)} - \theta^*\| + 4 \sum_{j \in [K]} \min_{\lambda_j \in \Lambda_j} \|\hat{\theta}_{I_j}^{(\lambda_j, I_j^c)} - \theta^*_{I_j}\| \; + \; 8\sqrt{2}\sigma \sum_{j \in [K]} \frac{\sqrt{\log|\Lambda_j|}}{\sqrt{n}} + \frac{8\sigma K x}{\sqrt{n}}$$

We now make explain the above theorem in more detail.

- The above theorem holds for all subgaussian error distributions. In the above theorem, the stated high probability event holds under the joint distribution of the errors ϵ and the (possibly) independently randomized assignment $I_1, \ldots, I_k$.

- Theorem 2.1 bounds the root sum of squared error (RSSE) of the cross-validated estimator $\hat{\theta}_{CV}$ as a sum of four terms. The first term is $\min_{\lambda \in \Lambda} \sqrt{SSE(\hat{\theta}^{(\lambda)}, \theta^*)}$. This is basically the RSSE of the optimally tuned version of $\hat{\theta}^{(\lambda)}$ as long as Λ is chosen to contain the theoretically optimal value of λ. Clearly, this term is necessary as the CV version has to incur RSSE atleast as much as what is incurred by the optimally tuned version. For instance, considering the example of Trend Filtering, state of the art bounds for the RSSE are known under appropriate choices of the tuning parameter (see [47], [45],[16]). As long as Λ is chosen containing these ideal choices of the tuning parameter; this term will scale exactly like the known bounds for Trend Filtering. The third term says that the dependence on the cardinality of Λ_j in the bound in Theorem 2.1 is logarithmic so as long as the cardinalities of Λ_j are bounded above by a polynomial in n our bound would only incur an additional $\log n$ term. It is not hard to ensure that the cardinality of Λ_j is at most a polynomial in n as will be shown in our applications. The fourth term gives a parametric $O(1/\sqrt{n})$ rate which is always going to be a lower order term.
- The second term appearing in the bound in Theorem 2.1 is really the key term which arises due to cross validation. Bounding this term becomes the central task in our applications. The second term behooves us, for each $j \in [K]$, to bound $CVERR_j(\lambda_j)$ for ***some good choice*** of the tuning parameter $\lambda_j \in \Lambda_j$.
- As per the earlier point, the main mathematical problem then facing us is to design completion estimators $\hat{\theta}^{(\lambda_j, I_j^c)}$ and bound the prediction errors $\min_{\lambda_j \in \Lambda_j} CVERR_j(\lambda_j)$. For instance, a first trivial step could be to write

$$\min_{\lambda_j \in \Lambda_j} CVERR_j(\lambda_j) = \min_{\lambda_j \in \Lambda_j} SSE(\hat{\theta}^{(\lambda_j, I_j^c)}_{I_j}, \theta^*_{I_j}) \leq \min_{\lambda_j \in \Lambda_j} SSE(\hat{\theta}^{(\lambda_j, I_j^c)}, \theta^*)$$

  where in the inequality we have just dropped the subscript I_j. Now, the problem of bounding the R.H.S $\min_{\lambda_j \in \Lambda_j} SSE(\hat{\theta}^{(\lambda_j, I_j^c)}, \theta^*)$ looks similar to the problem of bounding the SSE of the original estimator $\hat{\theta}^{(\lambda)}$ with one major difference. The estimator $\hat{\theta}^{(\lambda_j, I_j^c)}$ is a completion estimator, meaning that it is a function only of $y^c_{I_j}$ in contrast with the original estimator (optimally tuned) $\hat{\theta}^{(\lambda)}$ which is based on the full data. Nevertheless, we will show that for several estimation methods, there exists a way to divide the data into folds $I_1, I_2, \ldots, I_K$, design completion estimators $\hat{\theta}^{(\lambda_j, I_j)}$ for $j \in [K]$ so that the prediction errors $\min_{\lambda \in \Lambda_j} CVERR_j(\lambda_j)$ scale like the usual SSE (possibly with an extra multiplicative log factor) for the original estimation method with optimal tuning. In Sections 4, 3 we will propose some specific ways to do this for Trend Filtering and Dyadic CART respectively. A high level intuition why we can expect $\min_{\lambda_j \in \Lambda_j} SSE(\hat{\theta}^{(\lambda_j, I_j^c)}, \theta^*)$ to have same rates of convergence as $\min_{\lambda_j \in \Lambda} SSE(\hat{\theta}^{(\lambda)}, \theta^*)$ is the following. Observe that $\hat{\theta}^{(\lambda_j, I_j^c)}$ is based on $y^c_{I_j}$ which has on the order of $\frac{n}{K}$ data points if $I_1, I_2, \ldots, I_K$ is chosen to have roughly equal size. On the other hand, $\hat{\theta}^{(\lambda)}$ is based on the full dataset with n points. A good estimator based on $\frac{n}{K}$ representative samples should have the same rate of convergence as a good estimator based on all the n samples with at most worse constants (since K is a small constant).

REMARK 2.1. Theorem 2.1 is useful only when K is constant and not growing with n. So it is not useful for leave one out cross validation for instance where $K = n$. A different theory would be needed for that and we leave it as a topic for future research.

2.3. ***Why is our CV Estimator theoretically tractable?***. In this section we explain what makes our CV Estimator theoretically tractable. In particular, we give a proof sketch of Theorem 2.1.

Proof Sketch:

- **Step** 1**:** A simple argument (see Lemma A.1) shows

$$\|\hat{\theta}_{CV} - \theta^*\| \leq 2\|\tilde{\theta} - \theta^*\| + \min_{\lambda \in \Lambda} \|\hat{\theta}^{(\lambda)} - \theta^*\|.$$

  Therefore, this step reduces our problem to bounding the squared error of the intermediate estimator $\tilde{\theta}$.
- **Step** 2**:**
  We note that

$$\|\tilde{\theta} - \theta^*\|^2 = \sum_{j=1}^{K} CVERR_j(\hat{\lambda}_j) = \sum_{j=1}^{K} SSE(\hat{\theta}_{I_j}^{(\hat{\lambda}_j, I_j^c)}, \theta^*_{I_j}).$$

  Therefore, this step reduces our problem to bounding $SSE(\hat{\theta}_{I_j}^{(\hat{\lambda}_j, I_j^c)}, \theta^*_{I_j})$ for $j = 1$ say, as the same argument can be used for all $j \in [K]$.
- **Step** 3**:**
  At this point, we make the crucial observation that conditionally on the (possibly random) assignment of folds $I_1, \ldots, I_K$ and the noise variables $\epsilon_{I_j^c}$ (on all folds except the jth fold), the estimator $\hat{\theta}_{I_j}^{(\hat{\lambda}_j, I_j^c)}$ can be seen as a least squares estimator (see Step 5 in Section 2.1) over the finite set $\{\hat{\theta}_{I_j}^{(\lambda_j, I_j^c)} : \lambda_j \in \Lambda_j\}$. Note that this finite set becomes non random, once we condition on $I_1, \ldots, I_K$ and $\epsilon_{I_j^c}$. This is because in our framework both the estimator $\hat{\theta}^{(\lambda_j, I_j^c)}$ and the set Λ_j can only be functions of $y_{I_j^c}$. This allows us to use an oracle risk bound for a least squares estimator over a finite set (see Lemma A.2) to conclude a conditional high probability statement for all $x \geq 0$,

$$P\left(SSE(\hat{\theta}_{I_j}^{(\hat{\lambda}_j, I_j^c)}, \theta^*_{I_j}) \leq 2 \min_{\lambda_j \in \Lambda_j} \|\hat{\theta}_{I_j}^{(\lambda_j, I_j^c)} - \theta^*_{I_j}\| + 4\sqrt{2}\sigma\left(\sqrt{\frac{\log|\Lambda_j|}{n}}\right) + 4\frac{x}{\sqrt{n}} \middle| I_1, , \ldots, I_K, \epsilon_{I_j^c} \right) \geq$$

  $1 - 2\exp(-x^2/2\sigma^2)$.
- **Step** 4**:** Note that the probability on the R.H.S in the above conditional high probability statement does not depend on the conditioned variables. Hence, we realize that we can actually drop the conditioning in the above statement which, along with the display in Step 1, then furnishes the statement in Theorem 2.1.

Let's compare with the commonly used way of performing K fold CV where $\Lambda_j = \Lambda$ and

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \sum_{j=1}^{K} CVERR_j(\lambda) \tag{3}$$

There does not seem to be a way to invoke a conditional least squares estimator intepretation here as we are minimizing over the sum of prediction errors over all folds at once. In general, the above method seems harder to analyze. The fact that we first construct K optimized tuning parameters $\hat{\lambda}_j$ (one for each fold) minimizing the prediction error on each fold is critical to have this conditional least squares estimator interpretation which allows us to use least squares theory and makes the theoretical analysis tractable.

3. Dyadic CART.

3.1. ***Background and Related Literature***. The Dyadic CART estimator is a computationally feasible decision tree method proposed first in [12] in the context of regression on a two-dimensional grid design. This estimator optimizes a penalized least squares criterion over the class of *dyadic decision trees*. Subsequently, several papers have used ideas related to dyadic partitioning for regression, classification and density estimation; e.g see [30], [39], [2], [49].

The two main facts about Dyadic CART are

- The Dyadic CART estimator attains an oracle risk bound; e.g see Theorem 2.1 in [6]. This oracle risk bound can then be used to show that the Dyadic CART estimator is minimax rate optimal (up to small log factors) for several function classes of interest.
- The Dyadic CART estimator can be computed very fast by a bottom up dynamic program with computational complexity linear in the sample size, see Lemma 1.1 in [6].

These two properties of the Dyadic CART make it a very attractive signal denoising method. However, this oracle risk bound is satisfied only when a tuning parameter is chosen to be larger than a threshhold which depends on the unknown noise variance of the error distribution. In practice, an user is naturally led to cross validate this tuning parameter. To the best of our knowledge, there has been no rigorous study done so far on Dyadic CART when the tuning parameter is chosen by cross validation. Our goal here is to propose a cross validated version of Dyadic CART in general dimensions which retain the two properties stated above. We now set up notations, define the Dyadic CART estimator more precisely and state the existing oracle risk bound.

3.2. ***Notations and Definitions***. Let us denote the d dimensional lattice with N points by $L_{d,n} := \{1, \ldots, n\}^d$ where $N = n^d$. The lattice design is quite commonly used for theoretical studies in multidimensional nonparametric function estimation (see, e.g. [29]) and is also the natural setting for certain applications such as image denoising, matrix/tensor estimation. Letting θ^* denote the true signal, our observation model becomes

$$y = \theta^* + \epsilon,$$

where y, θ^*, ϵ are real valued functions on $L_{d,n}$ and hence are d dimensional arrays. Furthermore, ϵ is a noise array consisting of i.i.d subgaussian errors with an unknown subgaussian norm $\sigma > 0$.

For any $a < b \in \mathbb{Z}_+$, let us define the interval of positive integers $[a, b] := \{i \in \mathbb{Z}_+ : a \leq i \leq b\}$ where $\mathbb{Z}_+$ denotes the set of all positive integers. For a positive integer n we also denote the set $[1, n]$ by just $[n]$. A subset $R \subset L_{d,n}$ is called an *axis aligned rectangle* if R is a product of intervals, i.e. $R = \prod_{i=1}^{d} [a_i, b_i]$. Henceforth, we will just use the word rectangle to denote an axis aligned rectangle. Let us define a *rectangular partition* of $L_{d,n}$ to be a set of rectangles $\mathcal{R}$ such that (a) the rectangles in $\mathcal{R}$ are pairwise disjoint and (b) $\cup_{R \in \mathcal{R}} R = L_{d,n}$.

For a given rectangle $R \subset L_{d,n}$ and any $\theta \in \mathbb{R}^{L_{d,n}}$ let us denote the array obtained by restricting θ to R by θ_R. For a given array $\theta \in \mathbb{R}^{L_{d,n}}$, let $k(\theta)$ *denote the smallest positive integer k such that a set of k rectangles $R_1, \ldots, R_k$ form a rectangular partition of $L_{d,n}$ and the restricted array θ_{R_i} is a constant array.* In other words, $k(\theta)$ is the cardinality of the minimal rectangular partition of $L_{d,n}$ such that θ is piecewise constant on the partition.

3.2.1. ***Description of Dyadic CART***. Let us consider a generic discrete interval $[a, b]$. We define a *dyadic split* of the interval to be a split of the interval $[a, b]$ into two equal intervals. To be concrete, the interval $[a, b]$ is split into the intervals $[a, a - 1 + \lceil (b - a + 1)/2 \rceil]$ and

$[a + \lceil (b-a+1)/2 \rceil, b]$. Now consider a generic rectangle $R = \prod_{i=1}^{d} [a_i, b_i]$. A *dyadic split* of the rectangle R involves the choice of a coordinate $1 \le j \le d$ to be split and then the jth interval in the product defining the rectangle R undergoes a dyadic split. Thus, a dyadic split of R produces two sub rectangles R_1 and R_2 where $R_2 = R \cap R_1^c$ and R_1 is of the following form for some $j \in [d]$,

$$R_1 = \prod_{i=1}^{j-1} [a_i, b_i] \times [a_j, a_j - 1 + \lceil (b_j - a_j + 1)/2 \rceil] \times \prod_{i=j+1}^{d} [a_i, b_i].$$

Starting from the trivial partition which is just $L_{d,n}$ itself, we can create a refined partition by dyadically splitting $L_{d,n}$. This will result in a partition of $L_{d,n}$ into two rectangles. We can now keep on dividing recursively, generating new partitions. In general, if at some stage we have the partition $\Pi = (R_1, \ldots, R_k)$, we can choose any of the rectangles R_i and dyadically split it to get a refinement of Π with $k+1$ nonempty rectangles. *A recursive dyadic partition* (RDP) is any partition reachable by such successive dyadic splitting. Let us denote the set of all recursive dyadic partitions of $L_{d,n}$ as $\mathcal{P}_{\mathrm{rdp},d,n}$. Indeed, a natural way of encoding any RDP of $L_{d,n}$ is by a binary tree where each nonleaf node is labeled by an integer in $[d]$. This labeling corresponds to the choice of the coordinate that was used for the split.

For a given array $\theta \in \mathbb{R}^{L_{d,n}}$, let $k_{\mathrm{rdp}}(\theta)$ *denote the smallest positive integer* k *such that a set of* k *rectangles* $R_1, \ldots, R_k$ *form a recursive dyadic partition of* $L_{d,n}$ *and the restricted array* θ_{R_i} *is a constant array for all* $1 \le i \le k$. In other words, $k_{\mathrm{rdp}}(\theta)$ is the cardinality of the minimal recursive dyadic partition of $L_{d,n}$ such that θ is constant on every rectangular partition.

By definition, we have for any $\theta \in \mathbb{R}^{L_{d,n}}$,

$$k(\theta) \le k_{\mathrm{rdp}}(\theta).$$

We can now define the Dyadic CART estimator for a tuning parameter $\lambda > 0$,

$$\hat{\theta}^{(\lambda)} := \underset{\theta \in \mathbb{R}^{L_{d,n}}}{\arg\min} \left(\|y - \theta\|^2 + \lambda k_{\mathrm{rdp}}(\theta) \right). \tag{4}$$

Equivalently, we can also define $\hat{\theta}^{(\lambda)} = P_{S_{\hat{\pi}^{(\lambda)}}} y$, where $\hat{\pi}^{(\lambda)}$ is a data dependent partition defined as

$$\hat{\pi}^{(\lambda)} := \underset{\pi \in \mathcal{P}_{\mathrm{rdp},d,n}}{\arg\min} \left(\|y - P_{S_\pi} y\|^2 + \lambda |\pi| \right).$$

In the above, for any $\pi \in \mathcal{P}_{\mathrm{rdp},d,n}$, S_π denotes the subspace of $\mathbb{R}^{L_{d,n}}$ which consists of all arrays which are constant on every rectangle of π and P_{S_π} denotes the orthogonal projection matrix on that subspace. The discrete optimization problem in the last display can be solved by a dynamic programming algorithm in $O_d(N)$ time which makes fast computation of Dyadic CART possible.

3.2.2. ***Existing Oracle Risk Bound and its Implications***. We now state the oracle risk bound satisfied by the Dyadic CART estimator.

THEOREM 3.1. *[Theorem* 2.1 *in [6]]*

Suppose the error vector ϵ *is gaussian with mean* 0 *and covariance matrix* $\sigma^2 I$. *Then there exists an absolute constant* $C > 0$ *such that if we set* $\lambda \ge C\sigma^2 \log N$, *then we have the following risk bound*

$$\mathbb{E}\|\hat{\theta}^{(\lambda)} - \theta^*\|^2 \le \inf_{\theta \in \mathbb{R}^N} \left[3\,\|\theta - \theta^*\|^2 + 2\lambda\, k_{\mathrm{rdp}}(\theta) \right] + C\sigma^2.$$

In the case when $d = 2$ the above oracle risk bound had already appeared in the original paper [12], albeit up to an extra log factor. This oracle risk bound actually can be used to show that the Dyadic CART estimator adaptively attains near optimal rates of convergence for several function classes of interest. For instance, the above oracle risk bound was used in [12] to show that Dyadic CART is minimax rate optimal over several bivariate anistropic smoothness classes of functions. In [6], the above oracle risk bound was used to show that the Dyadic CART estimator is minimax rate optimal over the class of bounded variation signals in general dimensions and thus matches the known rates of convergence attained by the Total Variation Denoising estimator; see [22], [38]. It has been explained in detail in [6] how the above oracle risk bound (along with its fast computation) puts forward Dyadic CART as a computationally faster alternative to Trend Filtering and its multidimensional versions while essentially retaining (and even improving in some aspects) its statistical benefits.

The main question we consider here is the following.

Can a cross validated version of Dyadic CART still attain the oracle risk bound in Theorem 3.1?

To the best of our knowledge, the above question is unanswered as of now. We answer this question in the affirmative in this paper. Since our cross validated version of Dyadic CART will also satisfy a result very similar to the oracle risk bound as in Theorem 3.1 it will essentially inherit all the known results for the usual Dyadic CART mentioned above.

3.3. ***Description of the CVDCART estimator.*** We will follow our general scheme of defining cross validated estimators as laid out in Section 2.1. Let $\hat{\theta}^{(\lambda)}$ be the family of Dyadic CART estimators with tuning parameter $\lambda \geq 0$ as defined in (4).

1. Set $K = 2$.

2. We divide $L_{d,n}$ randomly into two folds/subsets I_1, I_2 as follows: Let $W \in \mathbb{R}^{L_{d,n}}$ be a random array consisting of i.i.d $Bernoulli(1/2)$ entries. Now define

$$I_1 = \{(i_1, \ldots, i_d) \in L_{d,n} : W(i_1, \ldots, i_d) = 1\}.$$

   The set I_2 is just the complement of I_1 in $L_{d,n}$.

3. Next, we define the estimators for $j \in \{1, 2\}$,

$$\hat{\theta}^{(\lambda, I_j)} = \underset{\theta \in \mathbb{R}^{L_{d,n}}}{\arg\min} ||y_{I_j} - \theta_{I_j}||^2 + \lambda k_{\text{rdp}}(\theta). \tag{5}$$

   Note that $\hat{\theta}^{(\lambda, I_j)}$ is a completion version of Dyadic CART because it only depends on y_{I_j}.

4. Consider an exponentially spaced finite grid of possible values of the tuning parameter λ_j, namely $\Lambda_j = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$, for $j \in \{1, 2\}$, where the choice of N^* will be determined later. Now define $\hat{\lambda}_j$ to be the candidate in Λ_j for which the prediction error $CVERR_j$ is the minimum, that is,

$$\hat{\lambda}_j := \underset{\lambda \in \Lambda_j}{\arg\min} \left|\left| y_{I_j} - \hat{\theta}^{(\lambda, I_j^c)}_{I_j} \right|\right|^2, \quad j \in \{1, 2\}.$$

   Now define an intermediate estimator $\tilde{\theta} \in \mathbb{R}^{L_{d,n}}$ such that

$$\tilde{\theta}_{I_j} = \hat{\theta}^{(\hat{\lambda}_j, I_j^c)}_{I_j}, \quad j \in \{1, 2\}. \tag{6}$$

5. Define

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} \|\hat{\theta}^{(\lambda)} - \tilde{\theta}\|^2$$

where $\Lambda = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$. Finally, our estimator (CVDCART) $\hat{\theta}_{CVDC}$ is defined such that

$$\hat{\theta}_{CVDC} = \hat{\theta}^{(\hat{\lambda})}. \tag{7}$$

3.4. ***Computation of the CVDCART estimator***. The major step in computing $\hat{\theta}_{CVDC}$ is to compute $\hat{\theta}^{(\lambda, I_j)}$ for $j = \{1, 2\}$. We present a lemma below stating the computational complexity of $\hat{\theta}_{CVDC}$.

LEMMA 3.2. *Let I denote the set I_1 or I_2. The computational complexity of the completion estimators $\hat{\theta}^{(\lambda, I)}$, i.e the number of elementary operations involved in computing $\hat{\theta}^{(\lambda, I)}$ is bounded by $C2^d dN$ for some absolute constant $C > 0$. Therefore, the overall computational complexity of $\hat{\theta}_{CVDC}$ is bounded by $CN^*2^d dN$. Since N^* can be taken to be $O(\log N)$ (as explained later), the overall computational complexity becomes $O_d(N \log N)$ in this case.*

REMARK 3.1. The above lemma ensures that our CVDCART estimator can also be computed in near linear time in the sample size.

In Section B, we describe a bottom up dynamic programming based algorithm to compute $\hat{\theta}^{(\lambda, I)}$. The underlying idea behind this algorithm is similar to the original algorithm given in [12] to compute the original version of Dyadic CART based on the full data. The description of the algorithm in Section B also clarifies what is the computational complexity of the algorithm, thereby proving Lemma 3.2.

3.5. ***Specification of*** $\hat{\theta}^{(\lambda, I)}$. There may be multiple solutions to the optimization problem defined in (5). For our main result (which is Theorem 3.3) to hold, we need to add one more specification which will complete the definition of $\hat{\theta}^{(\lambda, I)}$ for any given subset $I \subset L_{d,n}$. Below, we use the notation $\overline{y}_I$ to denote the mean of all entries of y in I.

In Section B, it is shown that the optimization problem in (5) can be solved by first solving the following discrete optimization problem over the space of all recursive dyadic partitions

$$\hat{\pi} = \arg\min_{\pi \in \mathcal{P}_{\mathrm{rdp},d,n}} \Big[\sum_{R \in \pi} 1(R \cap I \neq \emptyset) \sum_{u \in R \cap I} (y_u - \overline{y}_{R \cap I})^2 + \lambda |\pi| \Big] \tag{8}$$

where the notation $\sum_{R \in \pi}$ means summing over all the constituent rectangles R of π and $|\pi|$ denotes the number of constituent rectangles of the partition π.

It is then shown that a solution $\hat{\theta}^{(\lambda, I)}$ to the optimization problem (5) is a piecewise constant array over the optimal partition $\hat{\pi}$. For all $u \in R$, for each constituent rectangle R of $\hat{\pi}$ such that the set $R \cap I$ is non empty,

$$\hat{\theta}_u^{(\lambda, I)} = \overline{y}_{R \cap I}. \tag{9}$$

It is possible for a constituent rectangle R of the optimal partition $\hat{\pi}$ to not contain any data point from I, i.e, the set $R \cap I$ is empty. In that case, $\hat{\theta}^{(\lambda, I)}$ can take any constant value

within R and still be an optimal solution to the optimization problem (5). In such a case, for all $u \in R$, we set

$$\hat{\theta}_u^{(\lambda,I)} = \overline{y}_I \tag{10}$$

This fully specifies the estimator $\hat{\theta}^{(\lambda,I)}$ which is a valid completion version of the Dyadic CART estimator being a function of y_I only.

REMARK 3.2. Note that the above specification still does not mean that $\hat{\theta}^{(\lambda,I)}$ is uniquely defined. This is because $\hat{\pi}$ in (8) is not necessarily uniquely defined. However, as long as we take a solution $\hat{\pi}$ to the optimization problem in (8) and then construct $\hat{\theta}^{(\lambda,I)}$ satisfying both (9) and (10), Theorem 3.3 holds.

3.6. ***Main Result for the CVDCART estimator***. Now we state an oracle risk bound for our proposed CVDCART estimator in general dimensions. Before that, let us define the following quantities

$$R(\theta^*, \lambda) := \inf_{\theta \in \mathbb{R}^{L_{d,n}}} \left(3\|\theta - \theta^*\|^2 + 2\lambda k_{\mathrm{rdp}}(\theta)\right), \quad \text{and}$$

$$V(\theta^*) := \max_{u,v \in L_{d,n}} |\theta_u^* - \theta_v^*|.$$

THEOREM 3.3. *Fix any $\alpha \geq 1$ and any $\delta > 0$. There exists an absolute constant $C > 0$ such that if we set our grid $\Lambda_j = \Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ for $j = \{1, 2\}$, satisfying*

$$2^{N^*} > C\sigma^2 \log N,$$

then we have the following bound

$$\mathbb{E}\frac{1}{N}\|\hat{\theta}_{CVDC} - \theta^*\|^2 \leq \frac{C_3}{N}\left\{\left(R(\theta^*, C\sigma^2 \log n) + \alpha\sigma^2 \log N\right)\left(\frac{V(\theta^*)}{\sigma} + \sqrt{\log N}\right)^2 + \sigma^2 \log(4N^*/\delta)\right\}$$

with probability at least $1 - \delta - C_1 \log(V(\theta^*)\sqrt{N})N^{-C_2\alpha}$, where C, C_1, C_2, C_3 are absolute positive constants which may only depend on the underlying dimension d.

It is now worthwhile discussing some aspects of Theorem 3.3.

1. The above theorem basically ensures that the mean squared error (MSE) of our cross validated estimator $\hat{\theta}_{CVDC}$ can also essentially be bounded (up to additive and multiplicative log factors) by the desired factor $R(\theta^*, \sigma^2 \log N)$. The same bound holds for the optimally tuned version of Dyadic CART as is stated in Theorem 3.1. The only essential difference is that our bound contains an extra multiplicative factor $(\frac{V(\theta^*)}{\sigma} + \sqrt{\log N})^2$. The term $V(\theta^*)$ captures the range of the underlying signal. For realistic signals, the range should stay bounded. In these cases, this extra multiplicative factor would then be a logarithmic factor.

2. Theorem 3.3 ensures that our cross validated estimator $\hat{\theta}_{CVDC}$ (up to log factors) enjoys a similar oracle risk bound as the optimally tuned version of Dyadic CART. Therefore, the CVDCART estimator essentially inherits all the known statistical risk bounds for Dyadic CART. In particular, CVDCART estimator would be minimax rate optimal (up to log factors) for several function/signal classes of interest such as anisotropically smooth functions (see [12]), piecewise constant signals on arbitrary rectangular partitions when $d \leq 2$ and signals with finite bounded variation (see [6]).

3. The existing risk bound Theorem 3.1 says that the optimal tuning parameter choice is $C\sigma^2 \log N$ for some absolute constant $C > 0$. Theorem 3.3 holds as long as such a choice of λ is included in Λ. This suggests that we would like to have the grid of choices Λ to be dense enough so that we do not miss the optimal tuning value range. On the other hand, we would like to have a sparse grid Λ because the computational complexity scales like the cardinality of Λ times the complexity of computing one Dyadic CART estimator for a given λ.

Observe that our risk bound (in particular $R(\theta^*, \lambda)$) scales proportionally with λ. This implies that missing the optimal tuning value by a factor of 2 means that we pay at most 2 times the MSE of the ideally tuned version. This fact allows us to take a geometrically growing grid $\Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$. Selecting such a sparse grid Λ then has obvious computational benefits. The only disadvantage here is that N^* then becomes like a tuning parameter to be set by the user. However, in practice and in theory, this seems to be a minor issue. Theorem 3.3 holds if 2^{N^*} is larger than the theoretically recommended choice of $\lambda = C\sigma^2 \log N$. Hence, plugging in even a gross overestimate σ_{over} of σ and choosing $N^* = C'(\log\log N + \log \sigma_{over})$ for a large enough constant C' would suffice for any realistic value of σ. In our simulations we simply take $N^* = \log_2 N$.

4. Trend Filtering.

4.1. ***Background and Related Work***. Trend Filtering, proposed by [24], is a univariate nonparametric regression method that has become popular recently; see [43] for a comprehensive overview. For a given integer $r \geq 1$ and any tuning parameter $\lambda \geq 0$, the r^{th} order trend filtering estimator $\hat{\theta}^{(r)}_\lambda$ is defined as the minimizer of the sum of squared errors when we penalize the sum of the absolute r^{th} order discrete derivatives of the signal. Formally, given a data vector y,

$$\hat{\theta}^{(r)}_\lambda := \left(\underset{\theta \in \mathbb{R}^n}{\arg\min} \frac{1}{2}||y - \theta||^2 + \lambda n^{r-1} ||D^{(r)}\theta||_1 \right) \tag{11}$$

where $D^{(1)}\theta := (\theta_2 - \theta_1, \theta_3 - \theta_2, \ldots, \theta_n - \theta_{n-1})$ and $D^{(r)}\theta$, for $r \geq 2$, is recursively defined as $D^{(r)}\theta := D^{(1)}D^{(r-1)}\theta$. For any positive integer $r \geq 1$, let us now define the r^{th} order total variation of a vector θ as follows:

$$\mathrm{TV}^{(r)}(\theta) = n^{r-1} \|D^{(r)}(\theta)\|_1 \tag{12}$$

where $\|\cdot\|_1$ denotes the usual ℓ_1 norm of a vector.

REMARK 4.1. The n^{r-1} term in the above definition is a normalizing factor and is written following the convention adopted in the trend filtering literature; see for instance the terminology of canonical scaling introduced in [38]. If we think of θ as evaluations of a r times differentiable function $f : [0, 1] \to R$ on the grid $(1/n, 2/n \ldots, n/n)$ then the Riemann approximation to the integral $\int_{[0,1]} |f^{(r)}(t)|dt$ is precisely equal to $\mathrm{TV}^{(r)}(\theta)$. Here $f^{(r)}$ denotes the rth derivative of f. Thus, for natural instances of θ, the reader can imagine that $\mathrm{TV}^{(r)}(\theta) = O(1)$.

A continuous version of these trend filtering estimators, where discrete derivatives are replaced by continuous derivatives, was proposed much earlier in the statistics literature by [27] under the name *locally adaptive regression splines*. By now, there exists a body of literature studying the risk properties of trend filtering under squared error loss. There exists two strands of risk bounds for trend filtering in the literature focussing on two different aspects.

Firstly, it is known that for any $r \geq 1$, a well tuned trend filtering estimator $\hat{\theta}^{(r)}_{\lambda}$ attains a MSE bound $O\left(\frac{(TV^{(r)}(\theta^*))^{1/r}}{n}\right)^{2r/2r+1}$. This bound is minimax rate optimal over the space $\{\theta \in \mathbb{R}^n : \mathrm{TV}^{(r)}(\theta) \leq V\}$ for a given $V > 0$ and has been shown in [42] and [48] building on earlier results by [27]. A standard terminology in this field terms this $O(n^{-2r/2r+1})$ rate as the *slow rate*.

Secondly, it is also known that an ideally tuned Trend Filtering (of order r) estimator can adapt to $\|D^r(\theta)\|_0$, the number of non zero elements in the r^{th} order differences, *under some assumptions on* θ^*. Such a result has been shown in [16] (for the constrained version of Trend Filtering of all orders) and [31] (for the penalized version of Trend Filtering with $r \leq 4$). In this case, the Trend Filtering estimator of order r attains the near parametric $\tilde{O}(\|D^{(r)}(\theta)\|_0/n)$ rate which can be much faster than the $O(n^{-2r/2r+1})$ rate. Standard terminology in this field terms this as the *fast rate*.

The big problem is that the results described above are shown to hold only under theoretical choices of the tuning parameter. These choices depend on unknown problem parameters and hence cannot be directly implemented in practice. Moreover different tuning is needed to achieve slow or fast rates. A square root version of Trend Filtering was proposed by [33] to mitigate this issue. It has been shown that the ideal choice of the tuning parameter for the square root version does not depend on the noise variance. However, the tuning parameter still needs to be set to a particular unspecified constant (differently depending on whether slow or fast rates are desired) and thus does not really solve this problem.

Therefore, a version of Trend Filtering which chooses the tuning parameter in a data driven way and attains both slow and fast rates is highly desirable. This naturally leads us to consider cross validation. Practical usage of Trend Filtering almost always involves cross validation to choose λ; e.g, see [35]. However, no theoretical properties are known for a cross validated version of Trend Filtering. We attempt to fill this gap in the literature by proposing a cross validated version of Trend Filtering based on our general framework. Our goal here is to show that our cross validated version nearly (atmost up to log factors) attains the risk (both the slow rate and the fast rate) of the ideally tuned versions.

4.2. ***Description of the CVTF Estimator***. We will again follow our general scheme of defining cross validated estimators as laid out in Section 2.1. Fix any $r \geq 1$ and let $\hat{\theta}^{(r)}_{\lambda}$ be the family of r^{th} order Trend Filtering estimators with tuning parameter $\lambda \geq 0$ as defined in (11).

1. Set $K = r + 1$.
2. Divide $[n]$ deterministically into K disjoint index sets (ordered) $I_1, I_2, \ldots, I_K$ as follows. Let $n_0 = \lfloor \frac{n}{K} \rfloor$. Then for any $j \in [K]$, define
$$I_j = \{Kt + j \leq n : t = 0, 1, \ldots, n_0\}.$$
In words, data points K positions apart are placed into the same fold.
3. For all $j \in [K]$, define $\tilde{y}(I_j^c) \in \mathbb{R}^n$ by interpolating $y_{I_j^c}$ as follows:
$$\tilde{y}(I_j^c)_i = \begin{cases} y_i & \text{if } i \in I_j^c \\ \sum_{l=1}^{r} (-1)^{l+1} \binom{r}{l} y_{i+l} & \text{if } i \in I_j \text{ and } i + r \leq n \\ \sum_{l=1}^{r} (-1)^{l+1} \binom{r}{l} y_{i-l} & \text{if } i \in I_j \text{ and } i + r > n \end{cases}.$$
In words, $\tilde{y}(I_j^c)$ is defined in such a way that within the index set I_j^c it is same as y but for any index in I_j it is linearly interpolated from the neighbouring indices of y in I_j^c by a r^{th} order polynomial interpolation scheme.

4. Next, we define the completion estimators

$$\hat{\theta}^{(\lambda, I_j^c, r)} := \underset{\theta \in \mathbb{R}^n}{\arg\min} \frac{1}{2} \left\| \tilde{y}(I_j^c) - \theta \right\|^2 + \lambda n^{r-1} \left\| D^{(r)} \theta \right\|_1, \quad j \in [K].$$

Note that $\hat{\theta}^{(\lambda, I_j^c, r)}$ is a valid completion version as it is a function of $y_{I_j^c}$ only.

5. Consider an exponentially spaced finite grid of possible values of the tuning parameter λ, namely $\Lambda_j = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$. For any $j \in [K]$, define $\hat{\lambda}_j$ to be the candidate in Λ_j for which the prediction error is the minimum, that is,

$$\hat{\lambda}_j := \underset{\lambda \in \Lambda_j}{\arg\min} \left\| y_{I_j} - \hat{\theta}^{(\lambda, I_j^c, r)}_{I_j} \right\|^2, \quad j \in [K].$$

Now define an intermediate estimator $\tilde{\theta} \in \mathbb{R}^n$ such that

$$\tilde{\theta}_{I_j} = \hat{\theta}^{(\hat{\lambda}_j, I_j^c, r)}_{I_j}, \quad j \in [K]. \tag{13}$$

6. Define

$$\hat{\lambda} = \underset{\lambda \in \Lambda}{\arg\min} \|\hat{\theta}^{(r)}_\lambda - \tilde{\theta}\|^2$$

where $\Lambda = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$. Finally, our estimator (CVTF) $\hat{\theta}^{(r)}_{CVTF}$ is defined such that

$$\hat{\theta}^{(r)}_{CVTF} = \hat{\theta}^{(r)}_{\hat{\lambda}}. \tag{14}$$

REMARK 4.2. A different yet valid choice of Λ_j and $\hat{\lambda}_j$, Λ and $\hat{\lambda}$ is described in Section 7.3.1.

4.3. ***Main Results for the CVTF Estimator***. Below we state both the slow rate and the fast rate results for the proposed CVTF estimator.

THEOREM 4.1. *[Slow Rate]*
Fix any $r \geq 1$ and any $\delta > 0$. There exists a constant C_r only depending on r such that if we take our grid $\Lambda = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$ satisfying

$$2^{N^*} \geq C_r \sigma \big(n \log n\big)^{1/(2r+1)}$$

then we have the following bound with probability at least $1 - \delta$,

$$\mathbb{E} \frac{1}{n} \|\hat{\theta}^{(r)}_{CVTF} - \theta^*\|^2 \leq \frac{2 C_r V^*}{n^r} \left| D^{(r)} \theta^* \right|_\infty + C_r \sigma^2 \left(n^{-\frac{2r}{2r+1}} (V^* \log(n/\delta))^{\frac{1}{2r+1}} + \frac{N^* + \log(1/\delta)}{n} \right),$$

where $V^* = n^{r-1} \left\| D^{(r)} \theta^* \right\|_1 = \mathrm{TV}^{(r)}(\theta^*)$.

THEOREM 4.2. *[Fast Rate]*
Fix any $1 \leq r \leq 4$ and any $\delta > 0$. Let $s = \|D^{(r)} \theta^*\|_0$ and $\mathcal{S} = \{j : (D^{(r)} \theta^*)_j \neq 0\}$. Then $\mathcal{S}$ can be represented as $\mathcal{S} = \{t_1, \ldots, t_s\} \subseteq [n - r]$, where $1 \leq t_1 < \cdots < t_s \leq n - r$. Also, let $t_0 := 0$ and $t_{s+1} := n - r + 1$. Next, we define $n_i := t_i - t_{i-1}$, $i \in [s + 1]$ and $n_{\max} := \max_{i \in [s+1]} n_i$.

Define the sign vector $q^* \in \{-1, +1\}^s$ containing the signs of the elements in $(D^{(r)} \theta^*)_{\mathcal{S}}$, that is, for every $i \in [s]$, $q_i^* := sign(D^{(r)} \theta^*)_{t_i}$, and the index set

$$\mathcal{S}^{\pm} := \{2 \leq i \leq s : q_i^* q_{i-1}^* = -1\} \cup \{1, s + 1\}.$$

Suppose θ^* satisfies the following minimum length assumption, for a constant $c > 1$,

$$n_{\max} \leq c n_i \text{ and } n_i \geq r(r+2) \text{ for all } i \in \mathcal{S}^{\pm}.$$

Then there exists a constant C_r only depending on r such that if we take our grid $\Lambda = \{1, 2^1, 2^2, 2^3, \ldots, 2^{N^*}\}$ satisfying

$$2^{N^*} \geq C_r \sigma s^{-(2r-1)/2} \sqrt{n \log n}$$

then we have the following bound with probability at least $1 - \delta$,

$$\mathbb{E} \frac{1}{n} \|\hat{\theta}^{(r)}_{CVTF} - \theta^*\|^2 \leq \frac{2C_r s}{n} \left| D^{(r)} \theta^* \right|^2_{\infty} + C_r \sigma^2 \left(\frac{s}{n} \log n \log(n/\delta) + \frac{N^* + \log(1/\delta)}{n} \right).$$

We now make some remarks to explain certain aspects of the above theorems.

1. We have presented both our slow rate and the fast rate theorem following the notations and presentation style adopted by [31] in Theorem 1.1 in their paper. We have done this mainly because our proofs rely on the results developed by [31] and also to remain consistent with the existing literature. The two theorems above ensure that the $\hat{\theta}_{CVTF}$ estimator essentially attains the slow rate and the fast rate (both implied by Theorem 1.1 in [31]) known for an ideally tuned penalized Trend Filtering estimator. The main difference in both our bounds are the extra additive terms involving $|D^{(r)}\theta^*|_\infty$. However, as we explain below, this is typically a lower order term.

2. Both the bounds above involve the term $|D^{(r)}\theta^*|_\infty$. Note that under the canonical scaling where $V^* = O(1)$, we have $|D^{(r)}\theta^*|_\infty \leq \|D^{(r)}\theta^*\|_1 = O(n^{1-r})$. This means that the terms involving $|D^{(r)}\theta^*|_\infty$ in our bounds can again be considered to be of a lower order for all $r \geq 1$ under realistic regimes of V^*.

3. In light of the above two remarks, under the canonical scaling, the bound in Theorem 4.1 can be read as scaling like the near minimax rate $\tilde{O}(n^{-\frac{2r}{2r+1}})$ and the bound in Theorem 4.2 scales like the near parametric rate $\tilde{O}(|D^{(r)}\theta^*|_0 \, n^{-1})$ up to additive lower order terms. Thus, our bounds show that the CVTF estimator attains the slow rate and the fast rate, up to log factors, and hence does not suffer too much in comparison to ideally tuned trend filtering estimators, atleast in the context of rates of convergence.

4. We only state Theorem 4.2 for $r \in \{1, 2, 3, 4\}$ and the assumptions on θ^* in Theorem 4.2 are identical to the assumptions made in Theorem 1.1 of [31]. This is because our proof is based on the proof technique employed by [31], as explained in Section D.1. The fast rate result in [31] also is shown to hold for $r \in \{1, 2, 3, 4\}$. To the best of our knowledge, a complete proof of the fast rate for penalized trend filtering of order $r > 4$ is not yet available in the literature. In contrast, fast rates have been established for an ideally tuned constrained trend filtering of all orders; see [16]. It is possible to develop a cross validated version of the constrained trend filtering using our framework and show that it will then enjoy fast rates for all orders $r \geq 1$. However, in this paper we prefer considering the penalized version due to its popularity and computational benefits.

5. The assumption $n_{max} \leq c n_i$ for a constant $c > 1$ means that the length of each of the blocks in $\mathcal{S}^{\pm}$ are within a constant factor of each other. This kind of minimum length assumption is standard and is also known to be necessary for fast rates to hold; see Remark 2.4 in [16]. Note that such a minimum length assumption is needed only for the blocks in $\mathcal{S}^{\pm}$ and not for all blocks. For example, when $r = 1$, the blocks in $\mathcal{S}^{\pm}$ are either the

first and last constant pieces of θ^* or those constant pieces of θ^* which constitute a local maxima stretch or a local minima stretch.

6. The ideal choice of the tuning parameter (as shown in Theorem 1.1 in [31]) depends on whether we desire the slow rate or the fast rate. However, both these choices scale (with n) like n^α for some $0 < \alpha < 1$. Therefore, as long as 2^{N^*} is chosen to be larger than these idealized choices, both the theorems presented above will hold. By construction, Λ contains an exponentially growing grid which means that N^* can be chosen so that it grows logarithmically in n, σ. Therefore, in the regime where σ stays bounded away from ∞, the term $\sigma^2 \frac{N^* + \log(1/\delta)}{n} = O(\log n/n)$ appearing in both of the above theorems is a lower order term. In practice, one can choose $2^{N^*} = n$, for instance, which will satisfy the required condition for realistic sample sizes n and σ.

5. Singular Value Thresholding for Matrix Estimation. Singular Value Thresholding (SVT) is a fundamental matrix estimation and completion method; see [3], [11], [5]. It is known that Singular Value thresholding is an *all purpose* matrix estimation method and performs well in a wide variety of structured matrix estimation problems; see [5]. However, the existing guarantees for this estimator depend on a thresholding parameter being chosen to be larger than a cutoff value which depends on the noise variance. In practice, the choice of the threshold matters in regards to the finite sample performance of the SVT estimator; see Section 5 (simulations) in [9] where the authors were investigating the SVT estimator in the context of estimating Nonparametric Bradley Terry Matrices. Thus, it is of both theoretical and practical interest in using a cross validated version of the SVT estimator. To the best of our knowledge, a theoretical analysis of a CV version of SVT is not available in the literature. Our goal here is to demonstrate that our CV framework is well suited to develop a cross validated version of this fundamental estimator.

5.1. ***Background and Related Literature***. The literature on SVT is vast. For our purposes here, we will just consider one particular result known for an optimally tuned SVT. We will then develop a CV version of SVT and show that this particular result continues to hold for our CV version of SVT as well. We consider the basic denoising setting where we observe

$$y = \theta^* + \epsilon$$

where θ^* is an underlying $n \times n$ signal matrix and ϵ is a $n \times n$ noise matrix consisting of i.i.d subgaussian errors with unknown subgaussian norm σ. Consider the data matrix y and consider its singular value decomposition

$$y = \sum_{i=1}^{n} s_i u_i v_i^t.$$

Let $S_\lambda = \{i : |s_i| > \lambda\}$ be the set of thresholded singular values of y with threshold level $\lambda > 0$. Define the estimator

$$\hat{\theta}^{(\lambda)} = \sum_{i \in S_\lambda} s_i u_i v_i^t. \tag{15}$$

This is how a standard version of the SVT estimator is defined. Under this setting, the following lemma can be traced back at least to Lemma 3 in [40]. It is probable that this result is even older. We use the notation $\|M\|_{op}$ to denote the operator norm of a $n \times n$ matrix M.

LEMMA 5.1. *For any fixed $\eta > 0$, if the threshold λ is chosen such that $\lambda = (1+\eta)\tau$ with $\tau > \|\epsilon\|_{op}$ then we have the following inequality:*

$$\|\hat{\theta}^{(\lambda)} - \theta^*\|^2 \leq 8(1+\eta)^2 \left[\sum_{j=1}^{n} \min\{\tau^2, \sigma_j^2(\theta^*)\}\right],$$

where $\sigma_j(\theta^*)$ is the jth largest singular value (in absolute value) of θ^*.

REMARK 5.1. The above lemma is purely a deterministic inequality; there is no notion of randomness here.

The following is a standard bound on the maximum singular value of a random subgaussian matrix quoted from [46].

THEOREM 5.2 (Theorem 4.4.5 from [46]). *Let ϵ be an $n \times n$ matrix whose entries are independent mean 0 subgaussian random variables with subgaussian norm at most σ. Then there exists an absolute constant $C > 0$ such that for any $t > 0$, we have*

$$P\left(\|\epsilon\|_{op} \leq C\sigma(\sqrt{n} + t)\right) \geq 1 - 2\exp(-t^2).$$

Combining Lemma 5.1 and Theorem 5.2 (after plugging in $t = \sqrt{n}$) immediately yields the following theorem.

THEOREM 5.3. *There exists an absolute constant $C > 0$ such that if the threshold λ is chosen satisfying $\lambda = C\,\sigma\sqrt{n}$, then the following inequality holds with probability at least $1 - 2\exp(-n)$,*

$$\frac{\|\hat{\theta}^{(\lambda)} - \theta^*\|^2}{n^2} \leq \frac{C}{n^2}\left[\sum_{j=1}^{n} \min\{n\sigma^2, \sigma_j^2(\theta^*)\}\right]$$

This theorem reveals the adaptive nature of the SVT estimator. This is because the right hand side is a deterministic quantity which only depends on the true signal θ^*. Intuitively, this term can be thought of as describing the spectral complexity of θ^*. The above risk bound can be used to derive the rates of convergence of the SVT estimator for several different types of classes of matrices of interest. We mention two standard classes below. For more interesting matrix classes where SVT can be applied; see [5].

1. *Low Rank Matrices*: If θ^* has rank k, then $\sigma_j(\theta^*) = 0$ for $j > k$ and hence we obtain a bound on the MSE which is $C\frac{k\sigma^2}{n}$. It is well known that this is the minimax rate of estimation for the class of $n \times n$ matrices of rank k.

2. *Nonparametric Bradley Terry Matrices*: For a general structured class of matrices, one can typically show by an approximation theoretic argument that the singular values decay at a certain rate, even if they do not become exactly 0 as in the exact low rank case. For example, [40] showed that the right hand side in the above theorem scales like $\frac{C}{\sqrt{n}}$ for the class of $n \times n$ Nonparametric Bradley Terry Matrices. These matrices are monotone in both row and column, upto an unknown permutation and arise in modeling of pairwise comparison data; see [40], [9].

The important point to note here is that λ needs to be set proportional to $\sigma\sqrt{n}$ for the above theorem to hold. Since σ is typically unknown and the constant C is unspecified it is natural to cross validate over λ. Therefore, it would be highly desirable for a cross validated version (where the tuning parameter is chosen in a data driven way) of the SVT estimator to also satisfy a risk bound of the form given in Theorem 5.3. We propose such an estimator in the next section.

5.2. ***Description of the CVSVT estimator***. We will follow our general scheme of defining cross validated estimators as laid out in Section 2.1. Let $\hat{\theta}^{(\lambda)}$ be the family of Singular Value thresholding estimators with threshold parameter $\lambda \geq 0$ as defined in (15).

1. Set $K = 2$.
2. Divide $[n] \times [n]$ into I_1, I_2 randomly as follows. Each entry $(i, j) \in [n] \times [n]$ belongs to I_1 or I_2 with probability $1/2$ independently of other entries.
3. Let us denote I_1 by I and I_2 by I^c. Define the $n \times n$ binary matrix W which takes the value 1 on the entries in I and 0 elsewhere. Now define $\tilde{y}(I) \in \mathbb{R}^{n\times n}$ as follows:
$$\tilde{y}(I) = 2y \circ W$$
where $\circ$ denotes the operation of entrywise multiplication of two matrices of the same size. Similarly, define
$$\tilde{y}(I^c) = 2y \circ (1 - W).$$
Thus, $\tilde{y}(I), \tilde{y}(I^c)$ are matrices obtained by zeroing out entries of y (corresponding to entries in I or I^c) and then doubling it.
4. Next, we define the completion estimator $\hat{\theta}^{(\lambda,I)}$ to be the SVT estimator applied to the matrix $\tilde{y}(I)$ with threshold λ. Define $\hat{\theta}^{(\lambda,I^c)}$ similarly using the matrix $\tilde{y}(I^c)$. Note that by definition, $\hat{\theta}^{(\lambda,I)}$ is a function only of y_I and hence is a valid completion estimator.
5. Consider an exponentially spaced finite grid of possible values of the tuning parameter λ_j, namely $\Lambda_j = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$, for $j \in \{1, 2\}$. Now define $\hat{\lambda}_j$ to be the candidate in Λ_j for which the test error is the minimum, that is,
$$\hat{\lambda}_j := \operatorname*{arg\,min}_{\lambda\in\Lambda_j} \left|\left| y_{I_j} - \hat{\theta}^{(\lambda,I_j^c)}_{I_j} \right|\right|^2, \quad j \in \{1, 2\}.$$
Now define an intermediate estimator $\tilde{\theta} \in \mathbb{R}^{L_{d,n}}$ such that
$$\tilde{\theta}_{I_j} = \hat{\theta}^{(\hat{\lambda}_j,I_j^c)}_{I_j}, \quad j \in \{1, 2\}. \tag{16}$$
6. Define
$$\hat{\lambda} = \operatorname*{arg\,min}_{\lambda\in\Lambda} \|\hat{\theta}^{(\lambda)} - \tilde{\theta}\|^2,$$
where $\Lambda = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$. Finally, our estimator (CVSVT) $\hat{\theta}_{CVSVT}$ is defined such that
$$\hat{\theta}_{CVSVT} = \hat{\theta}^{(\hat{\lambda})}. \tag{17}$$

REMARK 5.2. The last two steps are same as for Dyadic CART and Trend Filtering. The main difference here is in the way we construct the completion estimator. We essentially construct $\tilde{y}(I)$ as an unbiased estimator of θ^* by randomly doubling or zeroing out each entry

of y and then perform SVT on $\tilde{y}(I)$ to create our completion version of the SVT estimator. This idea of randomly zeroing out and inflating other entries to preserve unbiasedness is not new and appears in several matrix completion papers. We call this particular method of creating completion estimators as the zero doubling method. This method is quite generic and can be used for several other signal denoising methods, see Section 7.2 for more on this.

5.3. ***Main Result.***

THEOREM 5.4. *Fix any $\delta > 0$. There exists an absolute constant $C > 0$ such that if we set our grid $\Lambda_j = \Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ for $j = \{1, 2\}$, satisfying*

$$2^{N^*} > C\left(|\theta^*|_\infty + \sigma\right)\sqrt{n},$$

then the following inequality holds with probability atleast $1 - 2\exp(-n) - \delta$,

$$\frac{\|\hat{\theta}^{(\lambda)} - \theta^*\|^2}{n^2} \leq \frac{C}{n^2}\left[\sum_{j=1}^{n} \min\{n\left(|\theta^*|_\infty + \sigma\right)^2, \sigma_j^2(\theta^*)\}\right] + \frac{C}{n^2}\sigma^2 \log(N^*/\delta).$$

We now make some remarks about this theorem.

REMARK 5.3. The above theorem ensures that our CVSVT estimator also enjoys the adaptive risk bound given in Theorem 5.3 with the only difference being that σ is replaced by the term $|\theta^*|_\infty + \sigma$. In realistic scenarios, the term $|\theta^*|_\infty$ should remain bounded even if n grows. Therefore, our CVSVT estimator essentially inherits all the implications of Theorem 5.3 for various structured subclasses of matrices.

REMARK 5.4. As mentioned before, setting N^* so that 2^{N^*} is larger than $C\left(|\theta^*|_\infty + \sigma\right)\sqrt{n}$ is a mild requirement as it is not hard usually to set upper bounds on the values of σ and $|\theta^*|_\infty$. Note that N^* would scale logarithmically in n, σ and $|\theta^*|_\infty$.

6. Lasso.

6.1. ***Background and Related Literature.*** The lasso, proposed by [41] is one of the most popular tools for high dimensional regression. By now, there is a vast literature on analyzing the mean squared error of lasso. The typical statement of the results say that if the tuning parameter λ is chosen appropriately depending on some problem parameters (which are typically unknown); then a certain MSE bound holds. However, in practice, the tuning parameter is often chosen using cross validation. The literature giving rigorous theoretical analysis of cross validated lasso is far thinner. As far as we are aware, the first few papers undertaking theoretical analysis of cross validated lasso are [25], [20], [19], [21], [28]. Two papers which contain the state of the art theoretical results on cross validated lasso are the papers [8], [10]. The paper [8] is the object of inspiration for the current article. They analyzed a two fold cross validated version of the constrained or primal lasso proposed in [41]. Their result gives the analogue of the so-called slow rate for Lasso (e.g, see Theorem 2.15 in [36]) in the fixed design setup. On the other hand, the paper [10] analyzes a related but different cross validated Lasso estimator and their main result gives an analogue of the fast rate for Lasso (e.g, see Theorem 2.18 in [36]) under random design with certain assumptions on the distribution of the covariates and the noise variables.

To the best of our knowledge, a single cross validated lasso estimator which attains both the slow rate and the fast rate in the fixed design setup has not yet been proposed in the literature. Our goal here is to demonstrate that designing such a cross validated lasso is possible.

We consider a two fold cross validated version $\hat{\beta}_{cvlasso}$ of the penalized lasso and prove two results. The first result, Theorem 6.1 gives the so-called slow rate under essentially no assumptions on the design matrix. This extends the result of [8] to cross validated penalized lasso. Our second result gives the fast rate for the same estimator $\hat{\beta}_{cvlasso}$ under a standard incoherence condition on the design matrix X. Thus, we are able to ensure that qualitatively both the results of [8] and [10] hold for our $\hat{\beta}_{cvlasso}$ estimator. We now describe the $\hat{\beta}_{cvlasso}$ estimator precisely. We consider a well specified linear model $y = X\beta^* + \epsilon$ where X is a fixed $n \times p$ design matrix and $\epsilon \in \mathbb{R}^n$ is an error vector consisting of i.i.d mean 0 subgaussian entries. We denote the standard lasso estimator with tuning parameter λ by $\hat{\beta}^{(\lambda)}$, defined as follows:

$$\hat{\beta}^{(\lambda)} := \underset{\beta \in \mathbb{R}^p}{\arg\min} \left[\sum_{i=1}^{n} (y_i - x_i^t \beta)^2 + \lambda \|\beta\|_1 \right],$$

where x_i^t is the ith row of the design matrix X.

6.2. ***Description of the CVLASSO estimator***.

1. Set $K = 2$.
2. Divide $[n]$ into I_1, I_2 randomly as follows. Each entry $i \in [n]$ belongs to I_1 or I_2 with probability $1/2$ independently of other entries.
3. For $j \in \{1, 2\}$, define

$$\hat{\beta}^{(\lambda, I_j)} := \underset{\beta \in \mathbb{R}^p}{\arg\min} \left[\sum_{i \in I_j} (y_i - x_i^t \beta)^2 + \lambda \|\beta\|_1 \right].$$

4. Consider a finite grid of possible values of the tuning parameter λ, namely $\Lambda = \{1, 2, 2^2, 2^3, \ldots, 2^{N^*}\}$ where N^* is chosen by the user. For any $j \in \{1, 2\}$, define $\hat{\lambda}_j$ to be the candidate in Λ for which the prediction error is the minimum, that is,

$$\hat{\lambda}_j := \underset{\lambda \in \Lambda}{\arg\min} \sum_{i \in I_j} (y_i - x_i^t \hat{\beta}^{(\lambda, I_j^c)})^2.$$

   Note that $I_1^c = I_2$ and vice-versa.
5. Now define an intermediate estimator $\tilde{\theta} \in \mathbb{R}^n$ such that for any $j \in \{1, 2\}$, if $i \in I_j$ then

$$\tilde{\theta}_i = x_i^t \hat{\beta}^{(\hat{\lambda}_j, I_j^c)}.$$

6. Define

$$\hat{\lambda} := \underset{\lambda \in \Lambda}{\arg\min} \|X\hat{\beta}^{(\lambda)} - \tilde{\theta}\|^2$$

   Finally, our estimator (CVLASSO) is defined to be

$$\hat{\beta}_{cvlasso} = \hat{\beta}^{(\hat{\lambda})}.$$

6.3. ***Main Results***.

THEOREM 6.1. *[Slow Rate] Suppose $M > 0$ is a number such that the design matrix X satisfies*

$$\max_{j \in [p]} \frac{1}{n} \sum_{i=1}^{n} X_{ij}^4 \leq M.$$

Fix any $0 < \delta < 1$. *Suppose we take our grid* $\Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ *such that*

$$2^{N^*} \geq 4\sqrt{2\sigma^2 M^{1/2} n \log p} + \sqrt{2\sigma^2 M^{1/2} n \log 1/\delta}.$$

Then we have the following bound with probability atleast $1 - 7\delta$ *for an appropriate absolute constant* $C > 0$,

$$\mathbb{E}\frac{1}{n}\|X\hat{\beta}_{cvlasso} - X\beta^*\|^2 \leq \frac{C}{n}\left[\|\beta^*\|_1^2\sqrt{Mn\log p} + \|\beta^*\|_1\sigma\sqrt{Mn\log p/\delta} + \sigma^2 N^* + \sigma^2 \log 1/\delta\right].$$

REMARK 6.1. The above result is qualitatively similar to the result in Theorem 2.1 in [8]. The main difference is that while that result is about a cross validated version of the constrained lasso, our result is about the corresponding cross validated version of the penalized lasso. There are certain advantages of using the penalized form of Lasso instead of the constrained form as mentioned in Remark 6.6.

REMARK 6.2. The bound in Theorem 6.1 basically says that the MSE of $\hat{\theta}_{cvlasso}$ scales like $O(\|\beta^*\|_1^2\sqrt{\frac{\log p}{n}})$ with high probability if M and σ are bounded away from ∞. Note that, this result holds essentially without any assumptions on the design matrix. As mentioned in [8], this MSE scaling agrees with the persistency condition for lasso (under random design) defined in [15] which says that if $\|\beta^*\|_1 = o\big((\frac{n}{\log p})^{1/4}\big)$ then persistency holds.

THEOREM 6.2. *[Fast Rate] Suppose that the design matrix* X *satisfies an incoherence condition*

$$\left|\frac{X^t X}{n} - I_{p\times p}\right|_\infty \leq \frac{1}{64k}$$

where $k = \|\beta^*\|_0$. *Suppose* $M > 0$ *is a number such that the design matrix* X *satisfies*

$$\max_{j\in[p]} \frac{1}{n}\sum_{i=1}^{n} X_{ij}^4 \leq M.$$

Assume that the sample size n *is large enough so that*

$$2\log p < \frac{n}{2^{14}k^2M^2}.$$

Fix any $0 < \delta < 1$. *Suppose we take our grid* $\Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ *such that*

$$2^{N^*} \geq 4\sqrt{2\sigma^2 M^{1/2} n \log p} + \sqrt{2\sigma^2 M^{1/2} n \log 1/\delta}.$$

Then we have the following bound with probability atleast $1 - 6\delta - \exp\left(-\frac{n}{2^{14}k^2M^2}\right)$ *for an appropriate absolute constant* $C > 0$,

$$\mathbb{E}\frac{1}{n}\|X\hat{\beta}_{cvlasso} - X\beta^*\|^2 \leq \frac{C}{n}\left[\frac{k^2}{\sqrt{n}}M(\log(p/\delta))^{3/2} + \sigma^2 k\sqrt{M}\log(p/\delta) + \sigma^2 N^* + \sigma^2\log 1/\delta\right].$$

REMARK 6.3. Assuming that M and σ are terms bounded away from ∞, the first term (inside the brackets) in the bound given in Theorem 6.2 which scales like $O\left(\frac{k^2}{\sqrt{n}}(\log p)^{3/2}\right)$ is dominated by the second term $O(k\log p)$ as long as $k < O(\sqrt{n/\log p})$ which we can readily check is the interesting regime where we can expect fast rates. This is because as soon as $k = O(\sqrt{n/\log p})$, the fast SSE rate $O(k\log p) = O(\sqrt{n\log p})$ which matches the slow rate. Therefore, when $k \geq O(\sqrt{n/\log p})$, one should use the slow rate result in Theorem 6.1. To summarize, the above result in Theorem 6.2 is useful in the sparse regime when $k < O(\sqrt{n/\log p})$ in which case the MSE of $\hat{\theta}_{cvlasso}$ scales like the fast rate $O(\frac{k\log p}{n})$ with high probability.

REMARK 6.4. In Theorem 6.2, we need a slightly stronger (by a factor of 2) incoherence condition than the standard one assumed for penalized Lasso in the literature. For example, Theorem 2.18 in [36] assumes that $\left|\frac{X^tX}{n} - I_{p\times p}\right|_\infty \leq \frac{1}{32k}$. It is well known that weaker (than incoherence) assumptions on the design matrix X such as the restricted isometry property are also sufficient to ensure fast rates for the lasso. Such results are likely to be true for our cross validated estimator $\hat{\theta}_{cvlasso}$ as well. However, we leave this for future research and just consider the incoherence condition because of two reasons. Firstly, it seems to be the simplest sufficient condition for fast rates available in the literature and is actually checkable in practice in contrast to the restricted isometry type properties which are computationally intractable to check. Secondly, our goal here is to simply demonstrate that both the slow rate and the fast rate are attainable for a single cross validated lasso estimator. Thus, we prefer to sacrifice some generality in exchange to demonstrate a phenomenon under a simpler sufficent condition.

REMARK 6.5. For both of our theorems, the grid $\Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ needs to satisfy that $2^{N^*} \geq C\sigma\sqrt{M^{1/2}n\log p}$ for a specified constant C. This is a very mild condition to ensure in practice. The parameter σ is the only unknown term and as explained before, even a gross over estimate can be plugged in without any serious consequences since our grid grows exponentially. Thus, the number of grid points would be $O(\log n \log\log p)$ which means we would need to solve the lasso optimization problem $O(\log n \log\log p)$ times to compute $\hat{\theta}_{cvlasso}$. In other words, the computational complexity of $\hat{\theta}_{cvlasso}$ would be $O(\log n \log\log p)$ times the computational complexity of computing a single instance of the penalized lasso estimator.

REMARK 6.6. A potential advantage of the cross validated penalized lasso over constrained lasso is as follows. It is known that under certain nonsingularity conditions on the design matrix X, to attain fast rates for the usual constrained lasso, it is sufficient that the tuning parameter is chosen to be exactly equal to $\|\beta^*\|_1$; see Theorem 2.1 in [4]. Clearly, this is hard to achieve in practice. It is not known to what extent is this result robust to the choice of this tuning parameter. For example, Theorem 2.1 in [4] further indicates that if the tuning parameter is chosen to be $\|\beta^*\|_1 \pm 1$ then it is not possible for the constrained lasso to attain fast rates. In contrast, the penalized lasso seems to be more robust with respect to the choice of its tuning parameter. For example, if one sets λ to be twice the ideal choice of λ known to achieve fast rates for penalized lasso (e.g, see Theorem 2.18 in [36]), then the risk at most doubles and hence the rate of convergence remains the same.

The upshot of this is that we can afford to have a grid of λ growing exponentially and still attain fast rates for our estimator $\hat{\theta}_{cvlasso}$. This has significant computational advantages as this means the cardinality of our grid Λ is only growing like $O(\log n \log\log p)$ which means we have to solve the lasso optimization problem at most $O(\log n \log\log p)$ times to compute $\hat{\theta}_{cvlasso}$. In contrast, it is likely that the Λ grid needs to be much finer in resolution (with cardinality growing like n^α for some $\alpha > 0$) for the cross validated constrained Lasso proposed in [8] to attain fast rates.

REMARK 6.7. Like in [8] we have proposed a 2 fold cross validated version of Lasso. However, if it is so desired, one can easily construct a similar K fold version as should be clear from our general framework and the description of $\hat{\theta}_{cvlasso}$. Similar risk bounds as in Theorem 6.1 and Theorem 6.2 would hold for the K fold version as well.

7. Discussion. In this section we discuss some naturally related matters.

7.1. *Other Signal Denoising Methods.* In this paper, we applied our CV framework to produce CV versions of Trend Filtering, Dyadic CART, Singular Value Threshholding and Lasso. Trend Filtering and Dyadic CART are the main focus of this paper. We considered Singular Value Threshholding and Lasso to further illustrate the generality of our framework. Our CV framework is based on a general principle and should be looked upon as providing a general recipe to develop theoretically tractable CV versions of potentially any other estimator which uses a tuning parameter. For example, using our framework, one should be able to develop theoretically tractable CV versions of the Total Variation Denoising estimator proposed by [37] (also see [22], [38], [7]), the Hardy Krauss estimator (see [13], [32]), the Optimal Regression Tree estimator proposed in [6], a higher dimensional version of Trend Filtering of order 2 proposed in [23] and potentially many more. As a start, the Zero Doubling method of constructing completion estimators alongwith using a geometrically doubling grid of candidate tuning values should be useable in these problems.

7.2. *Three Different Methods for Creating Completion Estimators.* One of the main ingredients of the CV framework proposed here is in the construction of the completion estimators. Basically, the user has to build a version of some estimator of interest which only depends on a subset of the data, namely y_I. In this paper, we have considered three different strategies for constructing these completion estimators. In the Dyadic CART and the Lasso examples, we basically restricted the squared error term in the optimization objective to only be summed over the subset I. Let us call this method *Restricted Optimization* (RO). In the Trend Filtering example, we first constructed an interpolated data vector $\tilde{y}(I) \in \mathbb{R}^n$ by interpolating on the subset of indices I^c using the values of y_I. We then fed this interpolated vector $\tilde{y}(I)$ into the Trend Filtering optimization objective. Let us call this method *Interpolate then Optimize* (IO). In the Singular Value Threshholding example, we zeroed out entries in I^c and doubled the entries in I to create a new matrix $\tilde{y}(I)$ and then used the singular value threshholding operator on this matrix $\tilde{y}(I)$. Let us call this method *Zero Doubling* (ZD). To summarize, RO, IO and ZD are really three different ways to construct completion estimators. In a given problem, the user can try any one of the three methods or even come up with a different method. For example, the ZD method is extremely generic and could have been used in the Trend Filtering or Dyadic CART examples as well. We did not do this because in our numerical experiments we found that RO was performing better (in MSE) than ZD (for Dyadic CART) by a factor of 2 or 3. Similarly, IO performed better than ZD in the case of Trend Filtering.

7.3. *Comparison with the R Package [1] Cross Validation Version of Trend Filtering.* It is instructive to compare the CVTF estimator proposed in this paper with the cross validation version of Trend Filtering implemented in the R package [1] (see the R command *cv.trendfilter*). In particular, it is worth noting the similarities and the differences in the two CV algorithms. Simulations comparing the finite sample performance of both these algorithms is given in Section 8.

1. We construct the folds in the same way as in the R package. Once the user decides the number of folds K, both methods choose K folds by placing every Kth point into the same fold. However, for rth order Trend Filtering in this paper we specifically set the number of folds to $K = r + 1$. We do this mainly to enable our interpolation scheme to create $\tilde{y}(I_j^c)$ for $j \in [K]$ which then allows us to obtain the two inequalities stated in Step 3 of the proof sketch in Section D.1.

2. In the R package implementation, the predictions for a given fold I_j is made by performing trend filtering on the shortened data vector $y_{I_j^c} \in \mathbb{R}^{|I_j^c|}$ and then the predicted value at a point is given by the average of the fits at this point's two neighbors (guaranteed to be in a different fold). We do this in the reverse order. We first interpolate by a polynomial interpolation scheme which is not the same as two neighbor averaging. We then apply trend filtering to this interpolated data vector $y(I_j^c) \in \mathbb{R}^n$ and obtain the completion estimators $\hat{\theta}^{(\lambda, I_j^c)}$. Our prediction at a point in I_j is just given by the fit $\hat{\theta}^{(\lambda, I_j^c)}$ at this point.
3. The main point of difference of the two methods is how they choose the final data driven value of the tuning parameter $\hat{\lambda}$. The R package implementation uses (3) to choose $\hat{\lambda}$ whereas our method is different as has been explained before.
4. The grid of candidate tuning values Λ_j and Λ are chosen in a fully data driven way in the R package implementation as explained in Section 7.3.1 below. We prefer to simply set $\Lambda_j = \Lambda = \{1, 2, 2^2, \ldots, 2^{N^*}\}$ for a large enough N^* such that $2^{N^*} = O(n)$. We could also mimick the R package implementation in choosing Λ_j and Λ and this will be perfectly in accordance with our CV framework as explained below in Section 7.3.1.

7.3.1. *An Alternative Way to Construct $\hat{\theta}^{(r)}_{CVTF}$.* Recall that from Step 5 onwards, the steps to construct the CVTF estimator is identical to the last few steps to construct the CVDCART estimator. However, unlike Dyadic CART, Trend Filtering is based on convex optimization which brings with it its inherent advantages. For the discussion below let us fix $r = 1$. Infact, (see [18]) for Trend Filtering of order 1 also known as Fused Lasso, the solution (as a function of λ) is piecewise constant with a finite number of pieces. Moreover, the entire path of solutions (for all $\lambda > 0$) can be computed in $O(n \log n)$ time and the number of distinct solutions is always bounded by $O(n)$.

The Rpackage implementing CV for Fused Lasso makes use of the above fact. The grid of candidate tuning parameter values in this package is simply taken to be a (finite) set of tuning values λ's which correspond to the set of all possible solutions.

We can mimick the Rpackage implementation in the last few steps and still stay within our CV framework which gives us a different CV version of Fused Lasso. This is because we can define Λ_j, for any $j \in [K]$, to be a finite set of tuning values, one for each of the distinct solutions of the following optimization problem

$$\min_{\theta \in \mathbb{R}^n} \frac{1}{2} \left\| \tilde{y}(I_j^c) - \theta \right\|^2 + \lambda n^{r-1} \left\| D^{(r)} \theta \right\|_1 .$$

This is allowed in our framework because this set only depends on $\tilde{y}(I_j^c)$ by definition. Similarly, we can define Λ to be a set of tuning values, one for each of the distinct solutions of the full optimization problem

$$\min_{\theta \in \mathbb{R}^n} \frac{1}{2} \left\| y - \theta \right\|^2 + \lambda n^{r-1} \left\| D^{(r)} \theta \right\|_1 .$$

Under these choices of Λ_j, Λ, our slow rate and fast rate theorems are still valid.

This is because, in view of Theorem 2.1, we would need to bound $\min_{\lambda \in \Lambda} SSE(\hat{\theta}^{(r)}_{\lambda}, \theta^*)$ and $\min_{\lambda \in \Lambda_j} SSE(\hat{\theta}^{(\lambda, I, 1)}_{I^c}, \theta^*_{I^c})$, where $I = I_j$ for $j \in [K] = [r+1]$. Note that $\min_{\lambda \in \Lambda} SSE(\hat{\theta}^{(r)}_{\lambda}, \theta^*) = \min_{\lambda \in \mathbb{R}} SSE(\hat{\theta}^{(r)}_{\lambda}, \theta^*)$ and thus we can use known bounds for the ideally tuned versions. Bounding $\min_{\lambda \in \Lambda_j} SSE(\hat{\theta}^{(\lambda, I, 1)}_{I^c}, \theta^*_{I^c})$ can be accomplished for this data driven choice of λ_j as well by again noting that $\min_{\lambda \in \Lambda_j} SSE(\hat{\theta}^{(\lambda, I, 1)}_{I^c}, \theta^*_{I^c}) = \min_{\lambda \in \mathbb{R}} SSE(\hat{\theta}^{(\lambda, I, 1)}_{I^c}, \theta^*_{I^c})$ and then simply following our existing proof. The only point to further consider would be the

term involving $\log|\Lambda_j|$ because now this is random. Here, we can invoke the result of [18] and use a deterministic bound (scaling like $O(n)$) on the cardinality of the random set $|\Lambda_j|$. Thus, the term involving $\log|\Lambda_j|$ can be bounded by $O(\frac{\log n}{n})$ term which is going to be a lower order term.

The advantage of this particular variant is that the entire procedure is fully data driven and one does not even need to set the value of N^* as before. This furnishes a truly completely data driven cross validated Fused Lasso estimator which attains both the slow rate and the fast rate. To the best of our knowledge, such a version of Fused Lasso did not exist in the literature before our work here.

One can also use this approach and choose Λ_j and Λ similarly, for Trend Filtering of any general order $r \geq 1$. This is because it is known (see Section 6.2 in [44]) that the entire path of solutions (for all $\lambda > 0$) can again be computed for Trend Filtering of any order $r \geq 1$. Moreover, the solution (as a function of λ) is piecewise linear and convex with a finite number of knots. So one can simply take Λ_j and Λ to be the finite set of knots of the appropriate optimization problems. This would then produce fully data driven CV (within our framework) versions of Trend Filtering of general order $r \geq 1$. To obtain a theoretical guarantee one can then use Theorem 2.1. The only missing part is that a deterministic bound on $\log|\Lambda_j|$ is not known for $r > 1$. However, from simulations we are led to conjecture that the number of distinct Trend Filtering solutions (of any order, w.r.t λ) grows at most polynomially with n. If this conjecture were true, we can then again conclude that the term involving $\log|\Lambda_j|$ is of a lower order term. We prefer to write our theorem for the current version because a) we have a complete proof of a risk bound for all orders $r \geq 1$, b) practically the parameter N^* is not hard to set and in our simulations both these versions perform similarly when we set $N^* = \log_2 n$.

7.4. *Heavy Tailed Errors.* The proof of our main result in Theorem 3.1 relies heavily on the errors being subgaussian. It would be interesting to explore the robustness of our CV framework to heavy tailed errors. In particular, can one develop CV versions of corresponding quantile versions of Dyadic CART and Trend Filtering (see [17] and [34]) using our framework? We leave this question for future research.

8. Simulations.

8.1. *Dyadic CART.* We conduct a simulation study to observe the performance of the proposed CV Dyadic CART estimator in three different scenarios each corresponding to a different true signal θ^*. In every case, the errors are generated from $N(0,1)$, the dimension $d = 2$ and we take $n = 128, 256, 512$. We estimate the MSE by 100 Monte Carlo replications and they are reported in Table 1. Overall, we see that our CV Dyadic CART estimator performs pretty well.

1. Scenario 1 [Rectangular Signal]: The true signal θ^* is such that for every $(i_1, i_2) \in L_{2,n}$, we have

$$\theta^*_{(i_1,i_2)} = \begin{cases} 1 & \text{if } n/3 \leq i_1, i_2 \leq 2n/3 \\ 0 & \text{otherwise} \end{cases}.$$

The corresponding plots are shown in Figure 1 when $n = 256$.

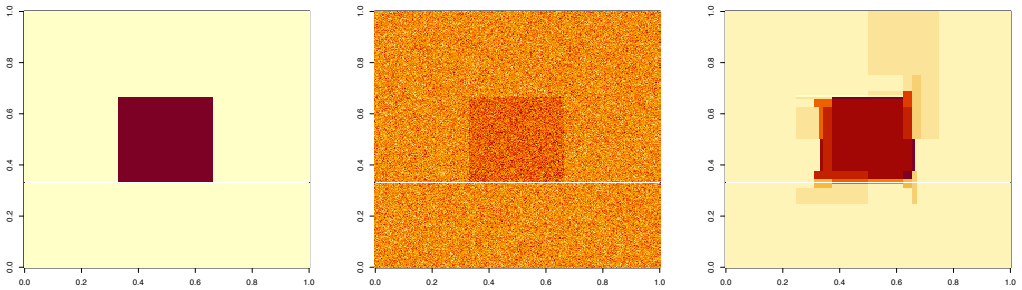

FIG 1. *The first diagram refers to the true signal, the second one to the noisy signal and the third one to the estimated signal by the CV Dyadic CART estimator.*

2. Scenario 2 [Circular Signal]: The true signal θ^* is such that for every $(i_1, i_2) \in L_{2,n}$, we have

$$\theta^*_{(i_1,i_2)} = \begin{cases} 1 & \text{if } \sqrt{(i_1 - n/2)^2 + (i_2 - n/2)^2} \leq n/4 \\ 0 & \text{otherwise} \end{cases}.$$

The corresponding plots are shown in Figure 2 when $n = 256$.

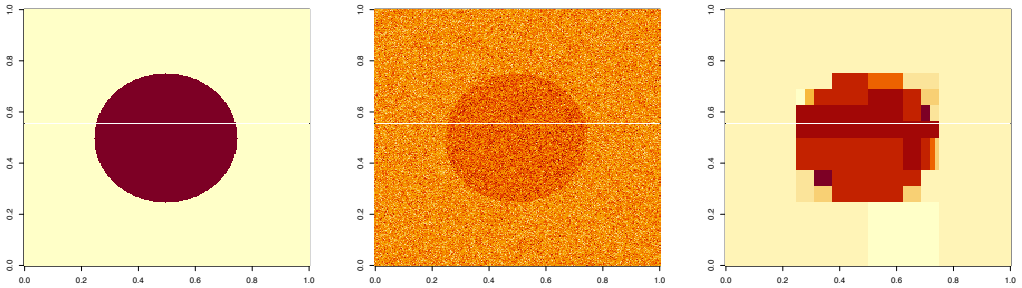

FIG 2. *The first diagram refers to the true signal, the second one to the noisy signal and the third one to the estimated signal by the CV Dyadic CART estimator.*

3. Scenario 3 [Smooth Signal]: The true signal θ^* is such that for every $(i_1, i_2) \in L_{2,n}$, we have $\theta^*_{(i_1,i_2)} = f\left(i_1/n, i_2/n\right)$, where

$$f(x, y) = 20 \exp\left(-5\{(x - 1/2)^2 + (y - 1/2)^2 - 0.9(x - 1/2)(y - 1/2)\}\right), \ 0 \leq x, y \leq 1.$$

The corresponding plots are shown in Figure 3 when $n = 256$.

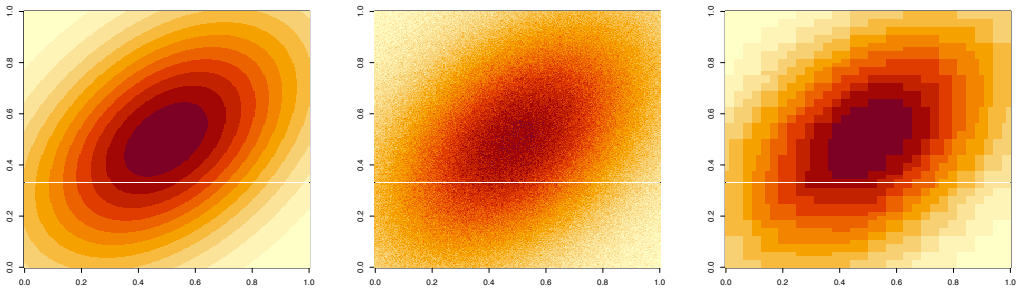

FIG 3. *The first diagram refers to the true signal, the second one to the noisy signal and the third one to the estimated signal by the CV Dyadic CART estimator.*

TABLE 1
MSEs of CV Dyadic CART estimator in different scenarios

n	Scenario 1	Scenario 2	Scenario 3
128	0.019	0.021	0.0005
256	0.005	0.014	0.0004
512	0.001	0.006	0.0003

8.2. *Trend Filtering.* We conduct a simulation study to observe the performance of the proposed CV Trend Filtering estimator and compare it to the CV version implemented in the R package *genlasso* [1]. The studies are carried out in four scenarios each corresponding to different true signals θ^*, where for any $i \in [n]$, we have $\theta^*_i = f(i/n)$ for some function $f : [0,1] \to \mathbb{R}$, specified below and the errors are generated from $N(0,1)$. In every scenario we consider the sample sizes $n = 300, 600, 1200, 2400$, and in each case, we estimate the MSE by 100 Monte Carlo replications. Furthermore, the comparison of MSEs in Scenarios 1, 2 and 3 are reported in Table 2.

1. Scenario 1 [Piecewise Constant Signal]: We consider the piecewise constant function

   $$f(x) = 2(1(x \in [1/5, 2/5])) + 1(x \in [2/5, 3/5]) - 1(x \in [3/5, 4/5]) + 2(1(x \in [4/5, 1])),$$

   and consider the Trend Filtering estimator of order $r = 1$. The corresponding plot is shown in the first diagram of Figure 4 when $n = 300$.
2. Scenario 2 [Piecewise Linear Signal]: We consider the piecewise linear function

   $$f(x) = 6x(1(x \in [0, 1/3])) + (-12x + 6)1(x \in [1/3, 2/3]) + (x - 8/3)(1(x \in [2/3, 1])),$$

   and consider the Trend Filtering estimator of order $r = 2$. The corresponding plot is shown in the second diagram of Figure 4 when $n = 300$.
3. Scenario 3 [Piecewise Quadratic Signal]: We consider the piecewise quadratic function

   $$f(x) = \begin{cases} 18x^2 & \text{if } x \in [0, 1/3] \\ -36(x - 1/2 - 1/\sqrt{12})(x - 1/2 + \sqrt{12}) & \text{if } x \in [1/3, 2/3] \\ 18(x-1)^2 & \text{if } x \in [2/3, 1] \end{cases}.$$

   and consider the Trend Filtering estimator of order $r = 3$. The corresponding plot is shown in the third diagram of Figure 4 when $n = 300$.

4. Scenario 4 [Smooth Sinusoidal Signal]: We consider the smooth sinusoidal function

$$f(x) = \sin 2\pi x + \cos 5\pi x,$$

and consider Trend Filtering estimators of order $r = 1, 2, 3$. The corresponding plots are shown in Figure 5 when $n = 300$.

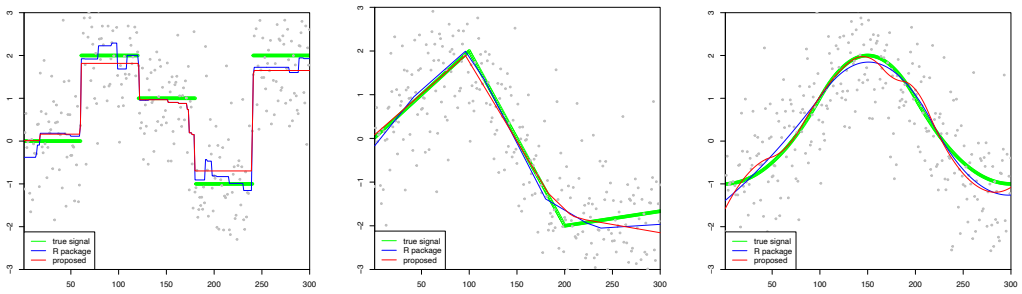

FIG 4. *The first diagram refers to the fits of order 1, the second one to fits of order 2 and the third one to the fits of order 3 in Scenarios 1, 2 and 3 respectively.*

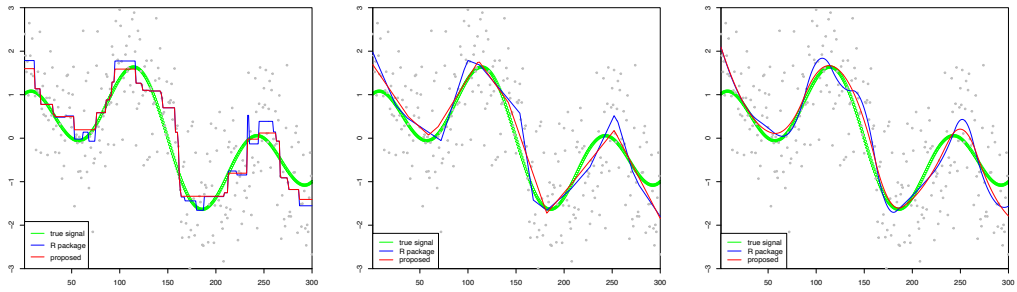

FIG 5. *The first diagram refers to the fits of order 1, the second one to fits of order 2 and the third one to the fits of order 3 in Scenario 4.*

TABLE 2
Comparison of MSEs between R package and CVTF in different scenarios.

	Scenario 1		Scenario 2		Scenario 3	
n	RPackage Fit	CVTF Fit	RPackage Fit	CVTF Fit	RPackage Fit	CVTF Fit
300	0.077	0.071	0.029	0.029	0.025	0.034
600	0.042	0.038	0.015	0.013	0.014	0.019
1200	0.023	0.022	0.007	0.006	0.007	0.006
2400	0.013	0.012	0.004	0.003	0.003	0.003

From Table 2, one can observe that the performance of our CV method is quite comparable to the existing R package and in fact, in most cases, the MSE of our method is slightly less. The results here are fully reproducible and our code is available upon request.

SUPPLEMENTARY MATERIAL

Supplement A : Supplementary File to "A Cross Validation framework for Signal Denoising with Applications to Trend Filtering, Dyadic CART and Beyond"
This supplementary contains the proofs of all the main results presented in this paper.

9. Funding. The second author is supported by NSF Grant DMS-1916375.

REFERENCES

[1] ARNOLD, T. B., TIBSHIRANI, R. J., ARNOLD, M. T. and BYTECOMPILE, T. (2020). Package 'genlasso'. *Statistics* **39** 1335–1371.

[2] BLANCHARD, G., SCHÄFER, C., ROZENHOLC, Y. and MÜLLER, K.-R. (2007). Optimal dyadic decision trees. *Machine Learning* **66** 209–241.

[3] CAI, J.-F., CANDÈS, E. J. and SHEN, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization* **20** 1956–1982.

[4] CHATTERJEE, S. (2014). A new perspective on least squares under convex constraint. *The Annals of Statistics* **42** 2340–2381.

[5] CHATTERJEE, S. (2015). Matrix estimation by universal singular value thresholding. *Annals of Statistics* **43** 177–214.

[6] CHATTERJEE, S. and GOSWAMI, S. (2019). Adaptive Estimation of Multivariate Piecewise Polynomials and Bounded Variation Functions by Optimal Decision Trees. *To appear in Annals of Statistics*.

[7] CHATTERJEE, S. and GOSWAMI, S. (2019). New Risk Bounds for 2D Total Variation Denoising. *arXiv preprint arXiv:1902.01215*.

[8] CHATTERJEE, S. and JAFAROV, J. (2015). Prediction error of cross-validated lasso. *arXiv preprint arXiv:1502.06291*.

[9] CHATTERJEE, S. and MUKHERJEE, S. (2019). Estimation in tournaments and graphs under monotonicity constraints. *IEEE Transactions on Information Theory* **65** 3525–3539.

[10] CHETVERIKOV, D., LIAO, Z. and CHERNOZHUKOV, V. (2020). On cross-validated lasso in high dimensions. *Annal. Stat.(Forthcoming)* **40**.

[11] DONOHO, D. and GAVISH, M. (2014). Minimax risk of matrix denoising by singular value thresholding. *Annals of Statistics* **42** 2413–2440.

[12] DONOHO, D. L. (1997). CART and best-ortho-basis: a connection. *The Annals of Statistics* **25** 1870–1911.

[13] FANG, B., GUNTUBOYINA, A. and SEN, B. (2021). Multivariate extensions of isotonic regression and total variation denoising via entire monotonicity and Hardy–Krause variation. *The Annals of Statistics* **49** 769–792.

[14] GOLUB, G. H., HEATH, M. and WAHBA, G. (1979). Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics* **21** 215–223.

[15] GREENSHTEIN, E., RITOV, Y. et al. (2004). Persistence in high-dimensional linear predictor selection and the virtue of overparametrization. *Bernoulli* **10** 971–988.

[16] GUNTUBOYINA, A., LIEU, D., CHATTERJEE, S. and SEN, B. (2020). Adaptive risk bounds in univariate total variation denoising and trend filtering. *The Annals of Statistics* **48** 205–229.

[17] HERNAN, O. and CHATTERJEE, S. (2021). Risk Bounds for Quantile Trend Filtering. *Biometrika*.

[18] HOEFLING, H. (2010). A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics* **19** 984–1006.

[19] HOMRIGHAUSEN, D. and MCDONALD, D. (2013). The lasso, persistence, and cross-validation. In *International Conference on Machine Learning* 1031–1039. PMLR.

[20] HOMRIGHAUSEN, D. and MCDONALD, D. J. (2014). Leave-one-out cross-validation is risk consistent for lasso. *Machine learning* **97** 65–78.

[21] HOMRIGHAUSEN, D. and MCDONALD, D. J. (2017). Risk consistency of cross-validation with lasso-type procedures. *Statistica Sinica* 1017–1036.

[22] HÜTTER, J.-C. and RIGOLLET, P. (2016). Optimal rates for total variation denoising. In *Conference on Learning Theory* 1115–1146.

[23] KI, D., FANG, B. and GUNTUBOYINA, A. (2021). MARS via LASSO. *arXiv preprint arXiv:2111.11694*.

[24] KIM, S.-J., KOH, K., BOYD, S. and GORINEVSKY, D. (2009). ℓ_1 trend filtering. *SIAM Rev.* **51** 339–360. MR2505584

[25] LECUÉ, G., MITCHELL, C. et al. (2012). Oracle inequalities for cross-validation type procedures. *Electronic Journal of Statistics* **6** 1803–1837.

[26] LI, Q. and RACINE, J. (2004). Cross-validated local linear nonparametric regression. *Statistica Sinica* 485–512.
[27] MAMMEN, E. and VAN DE GEER, S. (1997). Locally adaptive regression splines. *The Annals of Statistics* **25** 387–413.
[28] MIOLANE, L. and MONTANARI, A. (2018). The distribution of the lasso: Uniform control over sparse balls and adaptive parameter tuning. *arXiv preprint arXiv:1811.01212*.
[29] NEMIROVSKI, A. (2000). Topics in non-parametric statistics. *Ecole d'Eté de Probabilités de Saint-Flour* **28** 85.
[30] NOWAK, R., MITRA, U. and WILLETT, R. (2004). Estimating inhomogeneous fields using wireless sensor networks. *IEEE Journal on Selected Areas in Communications* **22** 999–1006.
[31] ORTELLI, F. and VAN DE GEER, S. (2019). Prediction bounds for (higher order) total variation regularized least squares. *arXiv preprint arXiv:1904.10871*.
[32] ORTELLI, F. and VAN DE GEER, S. (2020). Adaptive rates for total variation image denoising. *Journal of Machine Learning Research* **21** 247.
[33] ORTELLI, F. and VAN DE GEER, S. (2021). Oracle inequalities for square root analysis estimators with application to total variation penalties. *Information and Inference: A Journal of the IMA* **10** 483–514.
[34] PADILLA, O. H. M. and CHATTERJEE, S. (2021). Quantile Regression by Dyadic CART. *arXiv preprint arXiv:2110.08665*.
[35] POLITSCH, C. A., CISEWSKI-KEHE, J., CROFT, R. A. and WASSERMAN, L. (2020). Trend filtering–I. A modern statistical tool for time-domain astronomy and astronomical spectroscopy. *Monthly Notices of the Royal Astronomical Society* **492** 4005–4018.
[36] RIGOLLET, P. and HÜTTER, J.-C. (2015). High dimensional statistics. *Lecture notes for course 18S997*.
[37] RUDIN, L. I., OSHER, S. and FATEMI, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena* **60** 259–268.
[38] SADHANALA, V., WANG, Y.-X. and TIBSHIRANI, R. J. (2016). Total variation classes beyond 1d: Minimax rates, and the limitations of linear smoothers. In *Advances in Neural Information Processing Systems* 3513–3521.
[39] SCOTT, C. and NOWAK, R. D. (2006). Minimax-optimal classification with dyadic decision trees. *IEEE transactions on information theory* **52** 1335–1353.
[40] SHAH, N., BALAKRISHNAN, S., GUNTUBOYINA, A. and WAINWRIGHT, M. (2016). Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning* 11–20.
[41] TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.
[42] TIBSHIRANI, R. J. (2014). Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics* **42** 285–323.
[43] TIBSHIRANI, R. J. (2020). Divided Differences, Falling Factorials, and Discrete Splines: Another Look at Trend Filtering and Related Problems. *arXiv preprint arXiv:2003.03886*.
[44] TIBSHIRANI, R. J. and TAYLOR, J. (2011). The solution path of the generalized lasso. *The annals of statistics* **39** 1335–1371.
[45] VAN DE GEER, S. and ORTELLI, F. (2019). Prediction bounds for (higher order) total variation regularized least squares. *arXiv preprint arXiv:1904.10871*.
[46] VERSHYNIN, R. (2018). *High-dimensional probability: An introduction with applications in data science* **47**. Cambridge university press.
[47] WANG, Y.-X., SHARPNACK, J., SMOLA, A. and TIBSHIRANI, R. J. (2016). Trend filtering on graphs. *Journal of Machine Learning Research* **17** 1–41.
[48] WANG, Y.-X., SMOLA, A. J. and TIBSHIRANI, R. J. (2014). The Falling Factorial Basis and Its Statistical Applications. In *ICML* 730–738.
[49] WILLETT, R. M. and NOWAK, R. D. (2007). Multiscale Poisson intensity and density estimation. *IEEE Transactions on Information Theory* **53** 3171–3187.
[50] WONG, W. H. (1983). On the consistency of cross-validation in kernel nonparametric regression. *The Annals of Statistics* **11** 1136–1141.