Abstract. This paper is concerned with the application of thresholding to the estimation of possibly sparse single sequence data observed subject to noise. In such problems, accuracy can be greatly improved by selecting a threshold that adapts to the unknown signal strength. We set out a classification and regression tree approach aimed to partition a sequence of inhomogeneous strength into component homogeneous regions where we can independently set a locally adaptive threshold and thus improve estimation.

Our method places a mixture prior on each coefficient consisting of an atom of probability at zero and a symmetric probability density. The mixing weight is chosen via Empirical Bayes. The decision on whether a split should occur is based on a score test. Having selected the partitioning and obtained the local mixing weight for each region, estimation is carried out using the posterior median.

We evaluate the performance of our method in the single sequence case and for wavelet denoising on both simulated and real data. In the wavelet context we consider two alternative implementations, splitting the coefficients levelwise and splitting the original domain. Our method is cheap to compute and in numerical comparisons our method shows excellent performance when compared with current thresholding techniques.

Key words and phrases. Classification and regression tree; Empirical Bayes; Locally adaptive; Sparsity; Thresholding; Wavelet denoising.
1. Introduction

A common statistical problem is the estimation of a long, possibly sparse, data sequence which has been observed subject to noise. These data often have some spatial dimension with the underlying signal strength varying locally to some extent with position along the sequence. Specifically, suppose we have, after possible rescaling, observations \( X = (X_i)_{i \in I} \) satisfying
\[
X_i = \mu_i + \epsilon_i, \quad \text{for } i \in I
\]
where the \( \epsilon_i \) are independent \( N(0, 1) \) random variables. Furthermore, permit the \( d \)-dimensional subscript \( i \in I \) to have a natural ordering that in some sense denotes the location of the observed \( X_i \). This positional index \( i \) may, or may not, contain some information about the strength of the signal in the neighbourhood of \( i \). Examples where such situations might arise include image processing of astronomical data, estimation of activation on gene-chip arrays and wavelet denoising.

A natural and well-studied approach to this problem is the concept of thresholding. If \( |X_i| \) is larger than some threshold \( T \) then we presume it corresponds to a non-zero \( \mu_i \), which is estimated accordingly. Alternatively, if \( |X_i| < T \) then \( \mu_i \) is estimated as 0. The success of any such thresholding method relies crucially upon the selection of the threshold \( T \) with the appropriate value depending upon the sparsity of the underlying \( \mu = (\mu_i)_{i \in I} \). A sparse \( \mu \) requires a high threshold to remove the mainly spurious noise while conversely a dense \( \mu \) needs a low threshold to keep the signal.

Many suggestions for choice of the threshold \( T \) have been proposed ranging from simple universal thresholding (Donoho and Johnstone 1994) to the more advanced data-dependent false discovery rate (FDR, Abramovich et al. 2006) and EbayesThresh (Johnstone and Silverman 2004). The majority of these methods do not however automatically adapt to potential local variations in the signal sparsity with position \( i \), but instead apply a constant global threshold to all the data under consideration. While such a uniform threshold allows very good estimation if no local variability in signal strength exists, this is unrealistic for many data sets which are known to possess regions of stronger signal that are spatially distinct from weaker areas of signal. On data sets with heterogeneous regions such as these a global threshold is not appropriate and its application can be disastrous. Instead, we would hope that by splitting the signal into distinct homogeneous regions and applying a separate suitable threshold in each, reconstruction quality could be much improved.

Utilizing neighbouring information to inform us of the signal strength, and hence the appropriate threshold to apply, at a particular position \( i \) has previously been considered in the block thresholding procedures of Cai and Silverman (2001) and Cai (2002) but with somewhat limited success. In this paper we present a new technique, TreeThresh, which aims to incorporate the ideas of classification and regression trees (CART, Breiman et al. 1984) to identify and partition a possibly heterogeneous signal into component homogeneous regions. These sub-regions can then be thresholded separately using a hopefully more appropriate threshold for each than would be obtained with a global thresholding scheme. We show that if such heterogeneity exists within the signal, our method typically outperforms techniques currently available while if no such structure exists, TreeThresh is still comparable to these alternatives. We also demonstrate the additional benefit that our method provides in being able to visibly separate signal and images into distinct regions, thus allowing insight into the structure of the data of interest.

Our paper proceeds in the following manner. In section 2 we give a brief exposition of the EbayesThresh procedure on which our method is based. We then continue with section 3, which describes how these ideas can be incorporated into a tree splitting scheme. Asymptotic consistency of tree splitting in this general context is established in section 4. Section 5 considers the application of our TreeThresh algorithm to the problem of non-parametric regression using wavelets and in particular two distinct options for how one might wish to perform the partitioning — levelwise in the wavelet domain, or in the original domain. A simulation study is performed in section 6, comparing our method with current alternative thresholding techniques both for the case of single sequence thresholding and for the case of wavelet denoising. Finally, in section 7 we conclude with a brief overview of our work.

2. Thresholding

Our method builds upon the EbayesThresh method of Johnstone and Silverman (2004, 2005) which we briefly describe below. Consider our estimation problem as described in (1). Assume that the sequence \( \mu \) may be sparse. A very natural way to try and capture this possible sparsity as suggested by Abramovich et al. (1998) is to place an independent prior on each \( \mu_i \) consisting of the mixture
\[
f_{\text{prior}}(\mu_i) = (1 - w)\delta_0 + w\gamma(\mu_i)
\]
The loglikelihood of the observed $\mathbf{x}$ is then
\[
l(w) = \sum_{i \in I} \log \left( (1 - w)\phi(x_i) + wg(x_i) \right),
\]
where $g = \gamma \ast \phi$ is the convolution of the density $\gamma$ with the standard normal $\phi$. There are many possibilities for the choice of this $\gamma$, for instance $\mathcal{N}(0, \tau^2)$ or the quasi-Cauchy prior consisting of a continuous mixture of normals suggested in the original EbayesThresh paper of Johnstone and Silverman (2004). We set out all the algorithms for a general $\gamma$. However, based upon personal correspondence with these authors, in the simulation study we have only considered $\gamma$ to be a Laplace density with a fixed scale parameter, i.e. $\gamma(x) = 0.25 \exp \left( -0.5|\tau| \right)$.

The loglikelihood is maximized over the range $w \in [w_n^*, 1]$ where $w_n^*$ is the parameter value which would result in the universal threshold $T(w_n^*) = \sqrt{2\log n}$. The corresponding score function, which we will come back to in section 3, is
\[
s(w) = \sum_{i \in I} \frac{g(x_i) - \phi(x_i)}{(1 - w)\phi(x_i) + wg(x_i)} = \sum_{i \in I} \frac{\beta_i}{1 + w\beta_i},
\]
where
\[
(3) \quad \beta_i = \beta(x_i) = \frac{g(x_i)}{\phi(x_i)} - 1.
\]
The score function $s(w)$ is decreasing in $w$, making the maximum marginal likelihood (MML) estimate $\hat{w}$ cheap to compute.

The obtained MML estimate $\hat{w}$ allows us to tune our threshold $T(\hat{w})$ to the particular data set. Indeed, Johnstone and Silverman (2004) demonstrate that their method as described above performs well on a range of signals from very sparse to very dense as well as attaining optimal theoretical estimation error bounds.

It is important to note that the above method does not take into account the relative position $i$ of each observation and the same threshold and estimates would be obtained with any permutation of these indices. In many practical situations however, the location $i$ will contain some information about the local strength of the signal as many data sets consist of distinct regions of dense signal separated from much sparser areas. These different parts of the signal will require different thresholds for optimal estimation. With TreecThresh we attempt to address this shortcoming by permitting the data sequence to be split into disjoint regions according to their relative sparsity or otherwise to create subsections which have more homogeneous sparsity behaviour. This is achieved using the methodology of classification and regression trees as we explain below. Within each region, we can then separately apply the EbayesThresh procedure to give a thresholding rule which will adapt to local changes in signal strength.

### 3. Tree-based thresholding

In this section we will assume that the signal $X_i$ is heterogeneous, i.e. $w$ in (2) can take differing values for two indices $i_1 \neq i_2$. However we assume that we can partition the set of all indices $I$ into regions $P_1 \cup \ldots \cup P_d = I$, $P_i \cap P_j = \emptyset$, such that $\{X_i, \ i \in P_k\}$ is homogeneous in each region $P_k$, i.e.
\[
f_{prior}(\mu_i) = (1 - w(P_i))\delta_0 + w(P_i)\gamma(\mu_i), \quad \text{for} \ i \in P_k.
\]
These regions consist of consecutive indices, i.e. each region is of the form $P_k = \{i \mid a_i^{(P_k)} \leq i_i \leq b_i^{(P_k)}, \ i = 1, \ldots, d\}$.

The algorithm presented here is based on the idea of identifying such homogeneous rectangular regions. As an exhaustive search over all possible partitions is prohibitive, we propose to use a greedy “one step look-ahead” strategy of recursively partitioning the signal. At every step, we split a region into two new regions such that the homogeneity of the two new regions is maximized — provided the likelihood is sufficiently improved.

The cornerstone of the proposed algorithm is a test whether the signal in two regions $L$ and $R$ is homogeneous, i.e. whether the weight $w$ is the same for both regions $L$ and $R$. Consider the score test
for \( H_0 : w^{(L)} = w^{(R)} \). The score is
\[
s(w^{(L)}, w^{(R)}) = \left( \sum_{i \in L} \frac{\beta_i}{1 + w^{(L)} \beta_i}, \sum_{i \in R} \frac{\beta_i}{1 + w^{(R)} \beta_i} \right),
\]
where \( \beta_i \) is as in (3). The Fisher information is
\[
J(w^{(L)}, w^{(R)}) = \begin{pmatrix}
\sum_{i \in L} \frac{\beta_i^2}{(1 + w^{(L)} \beta_i)^2} & 0 \\
0 & \sum_{i \in R} \frac{\beta_i^2}{(1 + w^{(R)} \beta_i)^2}
\end{pmatrix}.
\]

Under \( H_0 \) the maximum likelihood estimates are \( \hat{w}_0^{(L)} = \hat{w}_0^{(R)} = \hat{w}^{(P)} \), where \( P = L \cup R \) and \( \hat{w}^{(P)} \) is the maximum-likelihood estimate of \( w \) in \( P \). Thus the statistic for the score test is
\[
\tau = s(\hat{w}^{(P)}, \hat{w}^{(P)})) J(\hat{w}^{(P)}, \hat{w}^{(P)})^{-1} s(\hat{w}^{(P)}, \hat{w}^{(P)})
\]
\[
= \frac{\left( \sum_{i \in L} \frac{\beta_i}{1 + \hat{w}^{(P)} \beta_i} \right)^2}{\sum_{i \in L} \left( \frac{\beta_i}{1 + \hat{w}^{(P)} \beta_i} \right)^2} + \frac{\left( \sum_{i \in R} \frac{\beta_i}{1 + \hat{w}^{(P)} \beta_i} \right)^2}{\sum_{i \in R} \left( \frac{\beta_i}{1 + \hat{w}^{(P)} \beta_i} \right)^2}
\]
\[
= \frac{\left( \sum_{i \in L} s_i^{(P)} \right)^2}{\sum_{i \in L} s_i^{(P)^2}} + \frac{\left( \sum_{i \in R} s_i^{(P)} \right)^2}{\sum_{i \in R} s_i^{(P)^2}},
\]
where
\[
s_i^{(P)} = \frac{\beta_i}{1 + \hat{w}^{(P)} \beta_i}.
\]

The statistic \( \tau \) of the score test is extremely economic to compute, provided the estimate \( \hat{w}^{(P)} \) is available, as it does not require the estimates \( \hat{w}^{(L)} \) and \( \hat{w}^{(R)} \) under the alternative. For this reason we can afford to consider all possible splits of \( P \) into \( (L, R) \) and to choose the one yielding the highest statistic \( \tau \). Other tests, like the Wald test or the likelihood ratio test, which are both asymptotically equivalent to the score test, would be unsuitable for this approach as they would be a lot more expensive to carry out. It should be noted that splits which would lead to too small regions are not considered since, after splitting, we require an estimate of \( w \) in each region that is robust. In our implementation the smallest region allowed was 10 observations though this choice was somewhat arbitrary. Regions where \( w \) is estimated to be either \( w_0^* \) or 1 are not split either: there is little merit in splitting a region already considered homogeneous.

This leads to the following algorithm:

1. Compute the \( \beta_i \) as in equation (3).
2. Start with a single region \( P = I \) and partition \( \mathcal{P} = \{I\} \).
3. Determine whether the region \( P \) should be split:
   i. For every admissible split of \( P \) into \( L \) and \( R \):
      Compute the score statistic \( \tau^{(L,R)} \) as in equation (4).
   ii. Select the best split \( (L^*, R^*) = \arg \max_{(L,R)} \tau^{(L,R)} \) leading to the largest score statistic.
   iii. Compute the maximum likelihood estimates \( \hat{w}^{(L^*)} \) and \( \hat{w}^{(R^*)} \) of the weights as well as the log-likelihoods \( l^{(L^*)} \) and \( l^{(R^*)} \) in the regions \( L^* \) and \( R^* \).
   iv. If \( 2(l^{(L^*)} + l^{(R^*)} - l^{(P)}) > \lambda_0 \) for some fixed \( \lambda_0 \), retain the split, i.e. update the partitioning
      \[ \mathcal{P} = \mathcal{P} \cup \{L^*, R^*\} \setminus \{P\} \]
      and carry out step 3 for both \( L^* \) and \( R^* \).

Note that the computation of the score statistic \( \tau^{(L,R)} \) can be sped up by caching the sums in the numerator and the denominator of the test statistic (4).

The recursive nature of the algorithm implies that the different regions created during the course of the algorithm can be arranged on a tree, whose leaves correspond to the partition obtained. From this point of view, the above algorithm can be seen as a special case of a classification and regression tree (CART, Breiman et al. 1984) using a special loss function.

This analogy to CART models can be exploited when deciding on the number of different regions to retain. Carrying out the above algorithm can yield too fine a partition corresponding to an overfit to the data. In this case, insufficient thresholding would be performed. On the other hand, selecting too small a \( \lambda_0 \) can lead to too coarse a partition. One possibility would be choosing \( \lambda_0 \) to be the corresponding quantile of the of the (asymptotic) distribution of the likelihood ratio statistic \( 2(l^{(L^*)} + l^{(R^*)} - l^{(P)}) \) or the score statistic \( \tau^{(L^*, R^*)} \). Note however that, due to the data-driven choice of \( L^* \) and \( R^* \), neither
of the two test statistics has the usual asymptotic $\chi^2$ distribution with one degree of freedom. More importantly, using $\lambda_0$ would lead to a short-sighted strategy for choosing the optimal tree: A seemingly worthless split might turn out to be an important boundary in a more complex partition.

Instead we propose to first estimate too fine a partition (by choosing too small a value of $\lambda_0$) and then "prune" it using the approach proposed by Breiman et al. (1984). Pruning corresponds to undoing (in a reversed order) some of the splits performed by the algorithm. It seems sensible to carry out the pruning in a way such that the penalized log-likelihood

$$l_P - \alpha \cdot |P|$$

is maximized for a certain value of $\alpha$. $l_P = \sum_{P\in\mathcal{P}} l(P)$ denotes hereby the log-likelihood obtained by partition $\mathcal{P}$ and $|P|$ denotes the number of regions in $\mathcal{P}$.

In complete analogy to the CART model one can show (see e.g. Ripley 1996, sec. 7.2) that there exists a nested sequence of partitions which maximize the penalized log-likelihood over different ranges of $\alpha$. Figure 1 illustrates this idea. This sequence of partitions and the corresponding sequence of $\alpha$’s can be found by directly applying Breiman et al.’s pruning algorithm to the tree created by the algorithm. Thus we only have to choose one element of the nested sequence of partitions, or, equivalently, choose the value of $\alpha$ that leads to the largest predicted log-likelihood. The cross-validation is carried out in exactly the same way as one would proceed for any CART-like model. (see e.g. Ripley 1996, pp. 224f for the algorithmic details).

4. THEORETICAL PROPERTIES

The following theorem establishes consistency for a large class of recursive partitioning methods. It does not make use of the specific way the best split is chosen in the algorithm presented in section 3.

**Theorem (Asymptotic Consistency).** Consider a $d$-dimensional array of $n_1 \times \cdots \times n_d = n$ observations

$$(X_i)_{i \in I}, \quad X_i = \mu_i + \epsilon_i,$$

with prior

$$f_{\text{prior}}(\mu_i) = (1 - w_i)\delta_0 + w_i\gamma(\mu_i),$$

where $\omega : [0,1]^d \to (0,1)$ is a continuous function and

$$w_i = \omega(i_1/n_1, \ldots, i_d/n_d), \quad \text{with } i = (i_1, \ldots, i_d).$$

Let $\mathcal{P}_n$ be a sequence of random partitions of the index space created by recursive splitting that may depend on the training sample. If

i. $\max_{P\in\mathcal{P}_n} \max_{i\in P} |w_i - \hat{w}(i)| \overset{P}{\to} 0$ for suitable $\hat{w}(P)$, and

ii. $\min_{P\in\mathcal{P}_n} |P|/\log n \overset{n}{\to} \infty$,

then as $n \to \infty$

$$\max_{P\in\mathcal{P}_n} \max_{i\in P} |\hat{w}(i) - w_i| \overset{P}{\to} 0,$$

where $\hat{w}(P)$ is the estimate of $\hat{w}$ in partition $P$.

Condition i. implies that the function $\omega(\cdot)$ can, in the limit, be approximated by a function that is constant in each region. This is the case if the (rescaled) size of each region converges to zero, or if the function $\omega(\cdot)$ is constant throughout a region. The second condition ii. implies that the number of observations in each region grows fast enough (faster than $\log n$); this is required in order to be able to find a consistent maximum likelihood estimator $\hat{w}(P)$ in each region.
It is easy to see that the proposed algorithm fulfills the conditions of the theorem if no pruning is carried out and the stopping rule based on the likelihood ratio statistic is replaced by a stopping rule based only on the size of a region (e.g. not performing splits yielding regions with less than \( n^a \) observations for some \( 0 < \gamma < 1 \)). In complete analogy with the results obtained for CART models (see for example the comments in Breiman et al. 1984, sec. 12), we cannot show that the pruning step and the stopping rule based on the likelihood ratio statistic satisfy the conditions of the theorem. Thus their justification lies more in their empirical performance than in the asymptotic theory.

The proof of the theorem can be found in appendix A. The key ingredient of the theorem is the limited capacity of the recursive partitioning approach: each region is a rectangle with \( 2d \) faces, thus there are less than \( n^2 \) possible regions. Note that the limited capacity is a necessary condition for consistency. If we were to choose “regions” without geometric constraints, there would be \( 2^n \) possible regions and we could for example define regions by ordering the observations according to \(|X_i|\). In this case our estimates of \( \hat{w}^{(P)} \) would not be consistent any more.

5. Thresholding of wavelet coefficients

Perhaps the most common current application of thresholding is non-parametric regression using wavelets. In addition, wavelet based thresholding creates an interesting question as to how one might wish to partition our space — in our original untransformed space or in coefficient space. For these two reasons we devote a short section explaining the application of the proposed TreeThresh algorithm to this problem.

Consider the standard non-parametric regression where we have \( 2^J \) noisy observations at regularly spaced points \( t_i \) of an unknown function \( f \),

\[
Y_i = f(t_i) + \epsilon_i
\]

where the \( \epsilon_i \) are independent \( N(0, \sigma^2) \) random variables. The standard wavelet approach involves transforming to the wavelet domain using the discrete wavelet transform (DWT) of Mallat (1989a,b) to obtain

\[
z_{j,k} = \theta_{j,k} + \epsilon_{j,k}
\]

where \( z_{j,k} \) is the DWT of the observed \( Y_i \); \( \theta_{j,k} \) the DWT of \( f(t_i) \); and due to orthogonality \( \epsilon_{j,k} \) are still independent \( N(0, \sigma^2) \) random variables. While this does not appear to have simplified the problem, a very large class of functions \( f \) can be approximated to a high degree of accuracy by relatively few wavelets. This expected parsimonious expansion, with most of the coefficients \( \theta_{j,k} \) close to zero, justifies thresholding to remove the noise component before transforming back to obtain the estimate (Donoho and Johnstone 1994).

Wavelet thresholding is usually performed separately for each level \( j \) of the wavelet coefficients, and indeed the simplest application of our method would be to apply the algorithm presented in the preceding section level-by-level. We call this approach Levelwise TreeThresh. One drawback of this Levelwise TreeThresh is that the partitions found for the different levels do not have to coincide, which impedes their interpretation. When thresholding wavelet coefficients it seems more natural to partition the original signal instead of partitioning the wavelet coefficients separately. A potential benefit of this approach is that it allows for sharing information about the segmentation across the different levels of the wavelet coefficients.

A function’s wavelet decomposition gives a multiresolution analysis in that each coefficient yields some information about \( f \) at a particular frequency and position. For a one-dimensional signal the coefficient \( \theta_{j,k} \) informs us about the behaviour of \( f \) near position \( 2^{-j}k \) on scale \( 2^{-j} \). As such splitting the signal at position \( i \) in the original domain would approximately correspond to splitting the wavelet coefficients for each level \( j \) at index \( k = 2^ji \). In the case of a two-dimensional signal, we can also allocate an analogous position for each coefficient and hence create an original domain splitting process for each coefficient in the decomposition where there are three matrices for each level which must be split coincidently. Figure 2 illustrates this idea.

This original domain splitting is simple to incorporate into the algorithm presented in section 3. We call this version Wavelet TreeThresh. As the wavelet coefficients are all independent, the score statistic for the joint partitioning of the wavelet coefficients is simply the sum over the score statistics \( \tau \) from equation (4) corresponding to the different levels. Even though a single partition is estimated, the weights \( w \) are estimated independently for the different levels and (in the case of dimension \( d > 1 \)) types of coefficient matrices.

One final consideration for Wavelet TreeThresh is how to carry forward narrow partitions as we pass down to coarser levels. As the number of wavelet coefficients shrinks as the level decreases, it seems
sensible to carry out less splits. For this reason, a split is only carried forward to a lower level if it would not yield a partition with fewer than \( n_0 \) wavelet coefficients, where \( n_0 \) is a predefined number (we used \( n_0 = 8 \)). Again, figure 2 demonstrates how this idea is applied to the wavelet decomposition of an image: the wavelet coefficients on the coarsest level are split into two partitions, whereas the wavelet coefficients at the finest level are split into five partitions.

Not carrying forward all splits means that the test statistics are not directly comparable since they correspond to tests with different degrees of freedom. This can be resolved by either using the corresponding p-value, or dividing the test statistic by the corresponding degrees of freedom. In the examples we have used the former approach.

6. Examples

6.1. Comparison with other methods. In order to judge the performance of our tree thresholding procedures it was compared to a variety of other recent thresholding techniques. The classical universal threshold (Donoho and Johnstone 1994) of \( \sqrt{2 \log n} \) aims to guarantee a noise free reconstruction. Alternatively, the FDR approach (Abramovich et al. 2006) uses principles from simultaneous hypothesis testing to control the false discovery rate. FDR requires pre-specification of a parameter \( q \) for which the authors do not provide a method, we have used \( q = 0.4 \). We have also included three block thresholding procedures which use neighbouring information to determine the thresholding at any particular point: BlockThresh (Cai 2002), NeighBlock and NeighCoeff (Cai and Silverman 2001). These techniques consider the data in blocks and threshold the data based upon the sum of squares within these blocks. Finally, we have also considered EbayesThresh (Johnstone and Silverman 2004), the technique upon which our TreeThresh is based. Further details can be found in the aforementioned papers.

For the wavelet thresholding illustrations, the wavelet transforms are computed using the WaveThresh software (Nason 1998) with the default wavelet choice of Daubechies’ least-asymmetric orthonormal compactly supported wavelet with 10 vanishing moments. Finally it should be noted that for our TreeThresh procedures we have retained without thresholding the scaling coefficients at coarse levels \( j \leq j_0 = 3 \) as is standard in the literature.

6.2. Basic Sequence Thresholding. To assess the performance of the basic TreeThresh algorithm presented in section 3 we performed a simulation study. We created sequences of 2000 \( \mu \)'s sampled from the mixture prior of section 2

\[
f_{\text{prior}}(\mu) = (1 - w)\delta_0 + w\gamma(\mu) \quad \text{with} \quad \gamma(x) = 0.25 \exp(-0.5|\gamma|)
\]
and $w = 0.01$ for $\mu_1, \ldots, \mu_{600}$, $w = 0.15$ for $\mu_{601}, \ldots, \mu_{1500}$, and $w = 0.03$ for $\mu_{1501}, \ldots, \mu_{2000}$. White noise $\epsilon_i \sim N(0, 1)$ was then added to create $x = (x_1, \ldots, x_{2000})$ as in (1), yielding a signal to noise ratio (SNR) of 0.88.

Our TreeThresh denoising procedure was applied to this observed $x$ with the number of partitions chosen via cross-validation and the estimated $\mu_i$ given by the posterior median of $\mu_i | X_i$ using the weight $w$ estimated by the algorithm. This was repeated for 1000 independent realizations of the positions of the splits. The two modes of the distribution of the positions of the splits correspond exactly to the cutoff points 600 and 1500. Figure 3(b) further shows that TreeThresh is more likely to create additional splits in areas where the weight $w$ is small, i.e. in areas where the signal is very sparse.

Figure 3(c) shows the distribution of the estimated weights $w_i$ for every index $i$. Once again we can see that our algorithm was able to pick up the change points rather accurately. However, there is a rather strong regression towards the global mean of the weights $w$. Towards the left and the right, $w$ is overestimated (the true $w$ is 0.01 and 0.03), whereas towards the middle, $w$ is underestimated (the true $w$ is 0.15). The reason for this is that in about 20% of the cases, a partition with too few regions was estimated, thus a single $\hat{w}$ was estimated for regions with different true $w$.

The average $l_2$ loss $\sum_{i=1}^n (z_i - \hat{z}_i)^2$ incurred by TreeThresh was 303.10 (with a standard error of 1.32), compared to an average $l_2$ loss of 317.30 (1.34) for EbayesThresh, 422.81 (1.79) for the universal threshold, 354.21 (1.52) when using the FDR ($q = 0.4$), 606.06 (2.49) for BlockThresh, 519.42 (2.14) for NeighBlock, and 436.82 (1.80) for NeighCoeff.

6.3. Wavelet thresholding. We now move on to analyze the performance of TreeThresh when applied to the problem of wavelet denoising of both one- and two-dimensional signals. We will consider both varieties of the splitting algorithm that we introduced: joint partitioning of the wavelet coefficients in TreeThresh; and the independent application of the basic TreeThresh algorithm to each level of the wavelet coefficients separately in Levelwise TreeThresh. The variance was estimated based on the mean absolute deviation of the coefficients on the finest level.

Table 1 gives the results obtained from 100 independent replications over the range of SNRs. In 12 of the 16 settings either Wavelet TreeThresh or Levelwise TreeThresh were the optimal methods offering significant improvements in $l_2$-loss when compared with the other methods. Meanwhile, in the four cases where they were outperformed, the difference between both TreeThresh algorithms and the best performing method is small. It does however not seem as if the joint thresholding used in the Wavelet TreeThresh offers any advantage over the Levelwise TreeThresh as far as the numerical performance is concerned. However it does offer the advantage of yielding a single interpretable partition which could be useful in applications where one is interested in determining which part of a signal may be worthy of further investigation.

Figure 4 also shows the signals reconstructed by the different methods for a single run with an SNR of $9 : 1$. The universal threshold $t^*$ is too large resulting in an overly smoothed estimate where important parts of the signal are lost. EbayesThresh performs less thresholding, which allows it to conserve more of the signal, however at the expense of removing less noise. The TreeThresh-based methods do not suffer from this problem as they can choose different thresholding schemes for different parts of the signal. Notably, they are the only methods able to reconstruct the fine scale oscillations at the start of Doppler. The Wavelet TreeThresh algorithm picks up all changes of the underlying signals, however the additional region in the middle of the Image signal only corresponds to noise. As mentioned in section 6.2 this is due to superfluous splits being carried out in areas where $w$ is small. As the results displayed in table 1 and figure 4 show, NeighBlock, NeighCoeff and BlockThresh yield very good results for the Doppler, but oversmooth the Image data.

Finally, we compare how the aforementioned methods perform when thresholding the wavelet coefficients of a two-dimensional image of $512 \times 512$ pixels. Figure 5 shows the original image used as well as the noisy image for an SNR of $3 : 1$ and the reconstructed signals. Table 2 gives the $l_2$ errors obtained from 100 replications. Once more, both variants of the TreeThresh algorithm clearly outperform the other methods. For large signal to noise ratios, TreeThresh offers a 40% reduction in the $l_2$ loss when compared with the next best method of EbayesThresh. Furthermore, visually Wavelet TreeThresh and
the Levelwise TreeThresh manage both to remove the noise from the sky and to maintain most of the details of the branches and leaves. Not being locally adaptive, the other algorithms either remove parts of the underlying signal or keep most of the noise. Panel (d) shows the partition found by the Wavelet TreeThresh algorithm; it gives a good representation of the main structure of the image. This illustrates the main benefit of the Wavelet TreeThresh algorithm; not only does it allow for denoising the image, it also yields a segmentation that corresponds to the different features of the image.

7. Conclusion

In this paper we have presented a new technique for thresholding which is able to adapt to potential local variation in signal/image strength. Being able to set a threshold dependent upon this strength is crucial to the success of estimation with unsuitable values resulting in extremely poor estimation.

TreeThresh incorporates principles of classification and regression trees (Breiman et al. 1984) into the EbayesThresh thresholding procedure (Johnstone and Silverman 2004). Our method aims to partition
wavelets. This latter approach has the advantage of identifying possible regions in the original image purely on a level-by-level basis. Alternatively, Wavelet TreeThresh attempts to partition the original performance when compared with current thresholding techniques. It was accurately and reliably able to the signal strength in that region can be applied.

Our method was initially implemented on single sequence data where it was shown to give excellent results for the one-dimensional examples. The method with the smallest average error $\sum_{i=1}^{n}(f_i - \hat{f}_i)^2$ (and its standard error) obtained for the image. The method with the smallest average error is highlighted with a box.

### Table 1

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SNR</th>
<th>Wavelet TT</th>
<th>Levelwise TT</th>
<th>EBayesThresh</th>
<th>Universal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 3 : 1</td>
<td>123.49 (1.60)</td>
<td>121.69 (1.38)</td>
<td>136.43 (1.08)</td>
<td>151.55 (1.20)</td>
<td></td>
</tr>
<tr>
<td>9 : 1</td>
<td>43.58 (0.50)</td>
<td>45.14 (0.49)</td>
<td>60.62 (0.61)</td>
<td>78.92 (0.68)</td>
<td></td>
</tr>
<tr>
<td>27 : 1</td>
<td>16.15 (0.20)</td>
<td>16.08 (0.18)</td>
<td>20.97 (0.20)</td>
<td>30.03 (0.25)</td>
<td></td>
</tr>
<tr>
<td>81 : 1</td>
<td>6.13 (0.07)</td>
<td>6.04 (0.06)</td>
<td>7.09 (0.06)</td>
<td>8.80 (0.09)</td>
<td></td>
</tr>
<tr>
<td>Doppler 3 : 1</td>
<td>32.97 (0.57)</td>
<td>30.80 (0.46)</td>
<td>33.00 (0.42)</td>
<td>36.52 (0.55)</td>
<td></td>
</tr>
<tr>
<td>9 : 1</td>
<td>12.67 (0.23)</td>
<td>11.69 (0.24)</td>
<td>15.21 (0.20)</td>
<td>16.48 (0.23)</td>
<td></td>
</tr>
<tr>
<td>27 : 1</td>
<td>4.69 (0.08)</td>
<td>4.37 (0.06)</td>
<td>5.83 (0.08)</td>
<td>6.39 (0.09)</td>
<td></td>
</tr>
<tr>
<td>81 : 1</td>
<td>1.69 (0.03)</td>
<td>1.35 (0.03)</td>
<td>1.98 (0.03)</td>
<td>2.03 (0.03)</td>
<td></td>
</tr>
<tr>
<td>Bumps 3 : 1</td>
<td>85.52 (0.77)</td>
<td>85.76 (0.71)</td>
<td>86.52 (0.74)</td>
<td>108.30 (0.95)</td>
<td></td>
</tr>
<tr>
<td>9 : 1</td>
<td>34.14 (0.34)</td>
<td>33.83 (0.32)</td>
<td>34.47 (0.35)</td>
<td>42.68 (0.45)</td>
<td></td>
</tr>
<tr>
<td>27 : 1</td>
<td>13.18 (0.11)</td>
<td>12.83 (0.11)</td>
<td>13.39 (0.10)</td>
<td>16.08 (0.12)</td>
<td></td>
</tr>
<tr>
<td>81 : 1</td>
<td>4.92 (0.04)</td>
<td>4.61 (0.03)</td>
<td>5.08 (0.04)</td>
<td>6.36 (0.04)</td>
<td></td>
</tr>
<tr>
<td>Blocks 3 : 1</td>
<td>64.22 (0.69)</td>
<td>64.00 (0.65)</td>
<td>63.99 (0.65)</td>
<td>76.67 (0.83)</td>
<td></td>
</tr>
<tr>
<td>9 : 1</td>
<td>30.69 (0.27)</td>
<td>30.57 (0.27)</td>
<td>30.88 (0.26)</td>
<td>36.98 (0.36)</td>
<td></td>
</tr>
<tr>
<td>27 : 1</td>
<td>11.60 (0.12)</td>
<td>11.36 (0.11)</td>
<td>11.70 (0.11)</td>
<td>14.95 (0.15)</td>
<td></td>
</tr>
<tr>
<td>81 : 1</td>
<td>4.19 (0.04)</td>
<td>4.02 (0.04)</td>
<td>4.24 (0.04)</td>
<td>5.38 (0.05)</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2

| Dataset | SNR | Wavelet TT | Levelwise TT | EBayesThresh | Universal | FDR |
|---------|-----|------------|--------------|--------------|-----------|
| Image 3 : 1 | 210.09 (3.01) | 228.30 (2.32) | 199.36 (2.22) | 168.64 (1.73) |
| 9 : 1 | 109.06 (1.37) | 115.93 (1.66) | 104.97 (1.42) | 87.89 (1.16) |
| 27 : 1 | 52.29 (0.79) | 42.82 (0.72) | 40.39 (0.71) | 34.79 (0.56) |
| 81 : 1 | 21.55 (0.32) | 13.40 (0.20) | 11.22 (0.19) | 10.19 (0.17) |
| Doppler 3 : 1 | 65.95 (1.08) | 43.20 (0.74) | 29.68 (0.53) | 26.35 (0.47) |
| 9 : 1 | 28.29 (0.43) | 15.50 (0.26) | 11.60 (0.21) | 13.22 (0.18) |
| 27 : 1 | 11.66 (0.16) | 5.36 (0.09) | 4.34 (0.07) | 4.41 (0.07) |
| 81 : 1 | 4.72 (0.06) | 1.97 (0.03) | 1.54 (0.02) | 1.58 (0.02) |
| Bumps 3 : 1 | 210.64 (1.92) | 154.51 (1.62) | 139.15 (1.51) | 120.75 (1.25) |
| 9 : 1 | 93.24 (1.14) | 61.55 (0.67) | 50.54 (0.57) | 41.78 (0.48) |
| 27 : 1 | 40.25 (0.37) | 21.16 (0.26) | 16.43 (0.22) | 13.72 (0.16) |
| 81 : 1 | 16.57 (0.16) | 7.44 (0.10) | 5.30 (0.07) | 4.71 (0.06) |
| Blocks 3 : 1 | 149.15 (1.69) | 101.56 (1.25) | 95.35 (1.11) | 85.74 (0.96) |
| 9 : 1 | 73.17 (0.71) | 47.12 (0.50) | 46.99 (0.48) | 47.13 (0.49) |
| 27 : 1 | 32.91 (0.30) | 22.28 (0.22) | 19.60 (0.22) | 15.19 (0.19) |
| 81 : 1 | 13.42 (0.16) | 7.79 (0.09) | 5.90 (0.08) | 4.35 (0.05) |

in inhomogeneous signals into distinct homogeneous component regions where individual thresholds suited to the signal strength in that region can be applied.

Our method was initially implemented on single sequence data where it was shown to give excellent performance when compared with current thresholding techniques. It was accurately and reliably able to identify where signals should be split and the resulting locally adaptive thresholds offered a significant improvement over the global EBayesThresh, FDR and universal thresholds.

For the problem of wavelet denoising we considered two differing implementations of our method. Levelwise TreeThresh considers each level of the wavelet decomposition independently and considers splits purely on a level-by-level basis. Alternatively Wavelet TreeThresh attempts to partition the original domain, with an implied domain, on each decomposition level, using the multi-resolution property of wavelets. This latter approach has the advantage of identifying possible regions in the original image which contain distinct characteristics and may be worthy of further investigation.
In our simulation study, using three standard test functions (Donoho and Johnstone 1994) and a real life two-dimensional image, both methods outperformed the well-regarded alternatives we considered. Of the two methods the Levelwise approach appeared to give the better numerical performance while Wavelet TreeThresh gave an interpretable partitioning which accurately distinguished the separate features of the image.
Figure 5. Noise-free image (a), noisy image (b, SNR 3:1) and reconstructions by the different methods
Appendix A. Proof of the theorem

Denote by $w^− = \inf\{\omega(\psi), \psi \in [0, 1]^d\} > 0$ and $w^+ = \sup\{\omega(\psi), \psi \in [0, 1]^d\} < 1$. Consider the first three derivatives of the loglikelihood:

$$s_1(w_1) = \frac{\partial f_1(w_1)}{\partial w_1}, \quad s_1'(w_1) = \frac{\partial^2 f_1(w_1)}{\partial w_1^2}, \quad s_1''(w_1), \quad s_1'''(w_1) = \frac{\partial^3 f_1(w_1)}{\partial w_1^3} = 2s_1(w_1)^3.$$

As $\beta_1 \in (-1, \infty)$ and $s_1(w_1) \in \left(\frac{\beta_1}{1 + w_1^\delta}, \frac{1}{w_1}w_1^\delta\right)$ with $\beta_1 = \frac{\beta(0)}{\delta(0)} - 1$, we can bound the moduli $|s_1(w_1)|, |s_1'(w_1)|,$ and $|s_1''(w_1)|$ by constants $\kappa, \kappa'$ and $\kappa''$.

Define $S_{P,n} = \sum_{P \in P} s(P, w)$. We then have

$$P(\max_{P} |S_{P,n}| > \varepsilon) = P\left(\sum_{P \in P} |s_i(w)\bigg|P\bigg| > \varepsilon\right) \leq 2 \exp(-\varepsilon/P/\kappa) \leq 2 \sum_{P \in S_{n,r}} \exp(-\varepsilon log^n/k) \leq n^2 \exp(-\varepsilon r log^n/k) = \exp(-\varepsilon r/kn^{2-k})$$

So far we considered the maximum over the set of all possible regions which grow faster than log $n$, we now turn to the maximum over all regions in the estimated sequence of partition $P_n$. Assumption ii. implies for all $r \in \mathbb{N}$ that $P(P_n \not\subset S_{n,r}) \longrightarrow 0$ as $n \rightarrow \infty$. Thus

$$P\left(\max_{P \in P_n} |S_{P,n}| > \varepsilon\right) \leq P\left(\max_{P \in P_n} |S_{P,n}| > \varepsilon\right) \cup \{P_n \not\subset S_{n,r}\} = P\left(\max_{P \in P_n} |S_{P,n}| > \varepsilon\right) \cap \{P_n \subset S_{n,r}\} + P\left(P_n \not\subset S_{n,r}\right) \longrightarrow 0,$$

i.e. $\max_{P \in P_n} |S_{P,n}| \overset{p}{\longrightarrow} 0$.

Now we obtain from the mean value theorem that

$$\max_{P \in P_n} \left|\sum_{P \in P} s_i(w(P))\bigg|P\bigg|\right| \leq \max_{P \in P_n} \left|\sum_{P \in P} s_i(w_1)\bigg|P\bigg|\right| + \max_{P \in P_n} \left|\sum_{P \in P} (w(P) - w_1) s'(w_1)\bigg|P\bigg|\right| \overset{P}{\longrightarrow} 0$$

Using the same approach as above one can show that $\max_{P \in P_n} \left|\sum_{P \in P} s_i'(w(P))\bigg|P\bigg|\right| \overset{P}{\longrightarrow} 0$ and that $E(s'(w_1))$ is in probability asymptotically bounded below by a positive real number. Together with the boundedness of the third (and higher order) derivatives of the loglikelihood this implies that

$$P\left(\bigcup_{P \in P_n} \{\text{Loglikelihood in region } P \text{ has a unique local maximum in } \hat{w}(P)\}\right) \overset{P}{\longrightarrow} 0,$$

which in turn implies

$$\max_{P \in P_n} |\hat{w}(P) - \hat{w}(P)| \overset{P}{\longrightarrow} 0.$$

Together with assumption i. this establishes the desired result.

References


Department of Mathematics, University of Bristol, University Walk, Bristol BS8 1TW, United Kingdom

E-mail address: l.evers@bris.ac.uk

Department of Probability and Statistics, University of Sheffield, Hicks Building, Hounsfield Road, Sheffield, S3 7RH, United Kingdom

E-mail address: T.Heaton@shef.ac.uk