

Rejoinder: Discussion of *Bump Hunting in High Dimensional Data*

Jerome H. Friedman & Nicholas I. Fisher

October 28, 1998

We thank all of the discussants for taking the time to contribute, and for both their compliments and helpful criticism. Before responding in detail, we address two general concerns that ran throughout many of the discussions.

1 Computation

Several discussants raised the issue of the computational requirements of the peeling algorithm (Section 7). They were concerned about its ability to scale to large (in memory) data sets. Here we provide some insight in the form of a worst case analysis. Let α be the minimum fraction of observations permitted to be trimmed from each box in the peeling sequence. The current version of the software (Friedman 1998) has been modified to incorporate such a limit for both real-valued and categorical variables. For the latter, this implies that more than one value may need to be trimmed from the current box at some iterations. This requires no additional computation since the values can be considered in the order of their individual output means within each current box.

Initially, before any peeling trajectories are constructed, the values of each real-valued variable are sorted in ascending order. The corresponding order information is stored as a linked list so that individual deletions can be performed later in constant time. This requires an initial computation proportional to $n_R N \log N$, where n_R is the number of real-valued variables and N is the training sample size.

An upper bound L on the number of boxes in each trajectory is given by

$$L = \log \beta_0 / \log(1 - \alpha)$$

where β_0 is the specified minimum box size (7.4). Here it is assumed that the minimum fraction α of observations is peeled at each iteration of the trajectory. In this case, the number of observations in the current box at the l th step is

$$N_l = N(1 - \alpha)^l.$$

For each categorical input variable the computation required to compute the peeling criterion (14.2) is proportional to N_l . For real-valued variables it is αN_l since the linked lists can be used, along with updating formulae for the means, so that only the α -fraction of the observations at the extremes need be considered. The total computation bound for producing a peeling trajectory in this case is proportional to

$$T = (n_R \alpha + n_C) N \sum_{l=1}^L (1 - \alpha)^l,$$

where n_C is the number of categorical input variables. A further upper bound is obtained by setting $L = \infty$ so that

$$T < (n_R + n_C / \alpha) N.$$

Including the initial sort the total computational order is bounded by

$$T \lesssim O(n_R N \log N) + O[CB(n_R + n_C/\alpha)N]$$

where C is the number of covering steps (Section 6), and B is the number of multiple trajectories (Section 13) computed for each one.

Examination of this expression shows that after the sort on the real-valued variables, the upper bound scales linearly with the training sample size N . It is also linear in the number of input variables $n = n_R + n_C$. For real valued variables, the upper bound is independent of the peeling fraction α and $n_R N$ represent a fairly tight bound on their contribution to it. For categorical variables, the upper bound is inversely proportional to α and $(n_C/\alpha)N$ represents a fairly loose bound; it is achieved only if every categorical variable selected for peeling has at least $1/\alpha$ distinct values in the boxes for which they were chosen. However, in situations for which there are many informative categorical variables with a large number of values, computation will tend towards the bound.

In his discussion, **Stone** reports running times for two data sets, both with $n = 17$, and with $N = 53560$ and $N = 717132$, respectively. Both sample size and reported compute time ratios were 13 so that computation increased linearly, lending some empirical support for the above analysis. Note that $B = 20$ multiple trajectories were used for these examples so that each single trajectory was computed in approximately 6 seconds for the smaller example, and 1.3 minutes for the larger one, on a 167MHz Sun UltraSPARC workstation. These times would be roughly halved on a typical 400MHz PentiumII PC.

2 Noise

Several of the discussants raised the issue of the performance of PRIM in highly noisy environments. In particular, the simulated example (Section 19.1) was criticized for the absence of noise on the output variable y . Even though the distribution of $y | \mathbf{x}$ was taken to be singular, there is still variance in the problem due to the randomness of the input variables \mathbf{x} . None the less, noiseless output variables are rare in data mining applications so here we revisit the example, introducing varying amounts of random noise on the output variable. In addition to performance considerations, the presence of noise raises issues of inference which were of concern to some discussants as well. Since there is very little discussion about inference in the paper, we take this opportunity to expand a little on it here.

2.1 Inference

Smyth illustrated that PRIM's search strategy has sufficient power to find bumps that are produced purely from sampling fluctuations generated from random data with constant $E[y | \mathbf{x}]$. This is more dramatically illustrated in Fig. 1. Here $N = 10000$ input vectors were sampled from a ten-dimensional uniform distribution, $\mathbf{x} \sim U^{10}[0, 1]$. The corresponding outputs were generated from a standard normal distribution, $y \sim N(0, 1)$, independent of \mathbf{x} . $B = 20$ bumps were used to increase the thoroughness of the search. The corresponding 20-trajectory plot was thinned (Section 13) so that only the undominated points (boxes) were retained. This whole procedure was repeated on ten such randomly generated data sets. Plotted in Fig. 1 are the collection of undominated points produced on these ten data sets based on a subsample of 6667 training observations for each one. (The three reference lines in Figures 1 and 2 are described below).

As seen in Fig. 1, PRIM was typically able to find 10%-support boxes (667 observations) with means of 0.2 output standard deviations, and three times that for boxes of 1% support (67 observations). Although these results confirm the power of the search strategy to find obscure bumps in complex output surfaces, they can be troubling from the perspective of inference, since all of these boxes have the same population mean value of zero; the apparent increased mean values of these boxes would not generalize to future data.

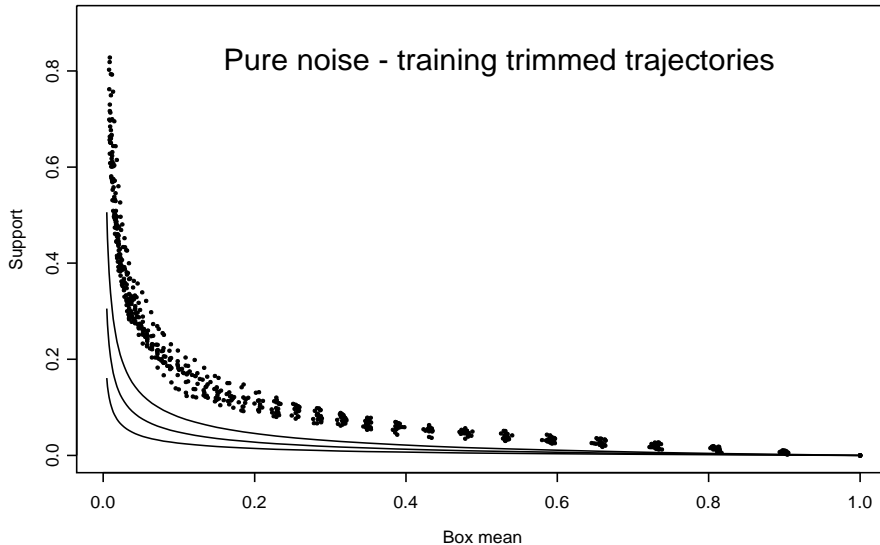


Figure 1: Undominated trajectory points obtained from ten *training* data sets for which $E[y|\mathbf{x}] = 0$. The outer curves bound a 95% confidence interval at each support value. The box means computed from the training data are seen to lie well above this interval, illustrating the high degree of overoptimism associated with validating on the training sample in highly noisy settings.

As discussed in the paper, cross-validation is suggested as a remedy for such “overfitting” caused by aggressive search strategies. For each peeling trajectory induced on the training data, the box means are estimated from the disjoint test subsample of 3333 observations. These are plotted for all 20 (bumped) trajectories, again removing the dominated points. The resulting plot is then used to judge mean-support trade-off.

Figure 2 shows the collection of undominated points obtained in this manner from the ten data sets represented in Fig. 1. One sees that the corresponding box mean estimates are substantially reduced, reflecting the fact that the test data were not used to induce the boxes. However, especially for small support values, these mean estimates are not centered at the population mean values of zero. This is a consequence of the point thinning; only the upper envelopes of undominated points over the ten trials are shown. The goal is to select the box of highest output mean for a given support; those with smaller mean at that support are not plotted since they would not be selected. The variance of the box mean estimates increases with decreasing support, thereby increasing their spread and thus their maximum values over the 20 bumped trajectories.

In general, using the test data for box selection is an important ingredient for increasing the power of the procedure. However, as noted by **Huber**, the test data then becomes another “training” sample, thereby clouding inference. This is a problem common to all procedures that use some form of cross-validation for model selection. The safest way to obtain a completely unbiased estimate of performance is to reserve a totally separate “validation” subsample that is not used either for induction or model selection. Straightforward inference can then be done on box means estimated from this validation sample.

One of the most fundamental inferential questions is whether or not there is any “signal” at all. That is, whether the obtained results actually reflect larger target function values in the selected boxes, or whether they are merely a reflection of aggressive training and model selection

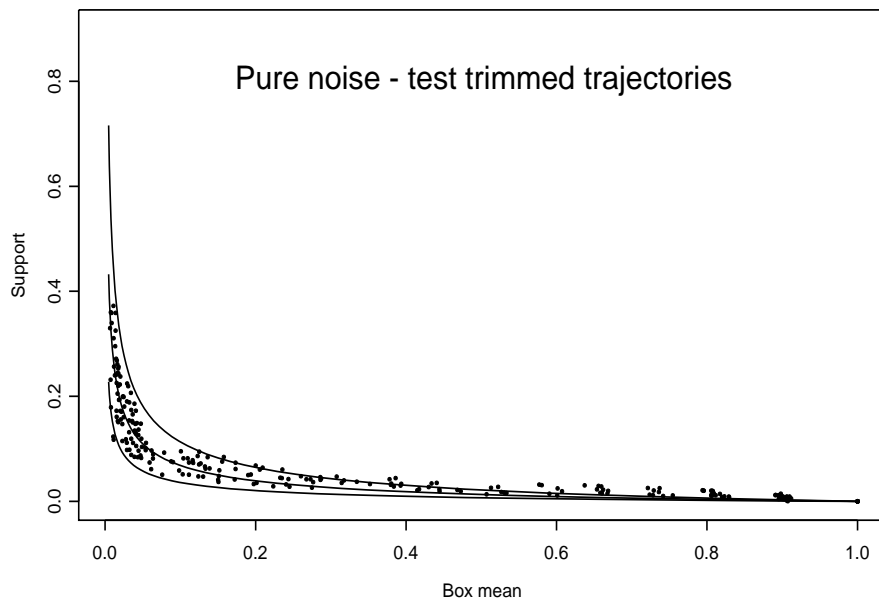


Figure 2: Undominated trajectory points obtained from ten *test* data sets for which $E[y|\mathbf{x}] = 0$. The outer curves bound a 95% confidence interval at each support value. The box means computed from the test data are seen to lie largely within this interval, except at small support values where there is substantial dependence among the bumped trajectories.

applied to pure noise. This is basically a hypothesis testing question, where the null hypothesis is that $f(\mathbf{x}) = E[y | \mathbf{x}]$ is constant, independent of \mathbf{x} . This can be judged using a validation sample as described above. However, it is possible to use the *test* sample for this, based on an approximate calculation. Under the assumptions that the box mean estimates are normally distributed, and that the B bumped trajectories are independent of each other, the distribution of the maximum (plotted) test sample box mean $m_{(B)}$ at each support value β is given by

$$F_B(m_{(B)} | \beta) = \Phi^B \left(m_{(B)} \sigma / \sqrt{\beta N} \right) \quad (\text{R1})$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function, N is the sample size, and σ^2 is the global variance of the output variable y .

The three curves in Fig’s 1 and 2 are respectively the 0.025, 0.5, and 0.975 quantiles of this distribution as a function of the support β . The outer two curves thus represent a 95% confidence interval for the upper envelope of the B induced trajectories under the null hypothesis and the assumptions. The distribution of the box means is usually quite close to being normal. Independence of the trajectories is more suspect, especially for large values of β near their beginning. The points derived from the test sample are seen in Fig. 2 to generally fall within the confidence interval for smaller support values ($\beta \lesssim 0.25$). For larger values of β , the upper envelope points fall above the interval due to the higher dependences there among the B induced trajectories from which they were derived. For the *training* data it is seen in Fig. 1 that all of the upper envelope points fall well above the corresponding null hypothesis interval owing to the high degree of over-fitting caused by estimating the box means on the same data used for inducing the boxes. This over-optimism is greatest in such pure noise situations. However for the *test* trajectories, as illustrated in Fig. 2, the quantile curves derived from (R1) can be used as a diagnostic to obtain a rough assessment as to whether points represent actual signal, or are just a reflection of searching on noise.

2.2 Power

We now turn to the effect that increasing the level of noise in the output variable has on the ability of PRIM to find high value regions of the underlying target function, $f(\mathbf{x}) = E[y | \mathbf{x}]$. As a test bed, we use the simulated example of Section 19.1. As noted, the distribution of $y | \mathbf{x}$ was there taken to be singular (19.1). Here we take this distribution to be normal

$$y | \mathbf{x} \sim N[f(\mathbf{x}), (\delta\sigma)^2]$$

where $\sigma^2 = E_{\mathbf{x}}[f^2(\mathbf{x})]$ is the variance of the target function over the (uniform) distribution of \mathbf{x} . Thus $1/\delta$ can be viewed as a signal-to-noise ratio. Here we take $\delta \in \{1, 2, 3, 4\}$ so that the underlying signal $f(\mathbf{x})$ accounts for respectively 50%, 33%, 25%, and 20% of the variance of the output variable y .

Figure 3 shows the first upper envelope trajectory plot produced by PRIM based on $B = 20$ bumped trajectories for each of the four noise levels. Because of the symmetry of the target (19.1) the first box is the most difficult to induce. The curves (R1) represent the null hypothesis that the entire output variance $(1 + \delta^2)\sigma^2$ is purely the result of noise. One sees that PRIM was able to produce meaningful (non-null) trajectories at all noise levels, with the $\delta = 4$ plot being somewhat marginal. For the first three noise levels $\delta \in \{1, 2, 3\}$ the high mean boxes all represented actual “solutions” in that they enclosed one of the true maxima of $f(\mathbf{x})$ (19.1) and the redundant variable elimination strategy (Section 10) unambiguously identified the first three input variables as the only relevant ones. For the highest noise level $\delta = 4$, the intervals on the first three variables always enclosed a maximum and they were identified as being the most relevant, but they could not be judged as being the only relevant variables defining the respective boxes.

Since this example is simulated it is possible to calculate analytically the population mean and support of any box. These are shown in Fig. 4 for selected boxes from each of the four

Your text

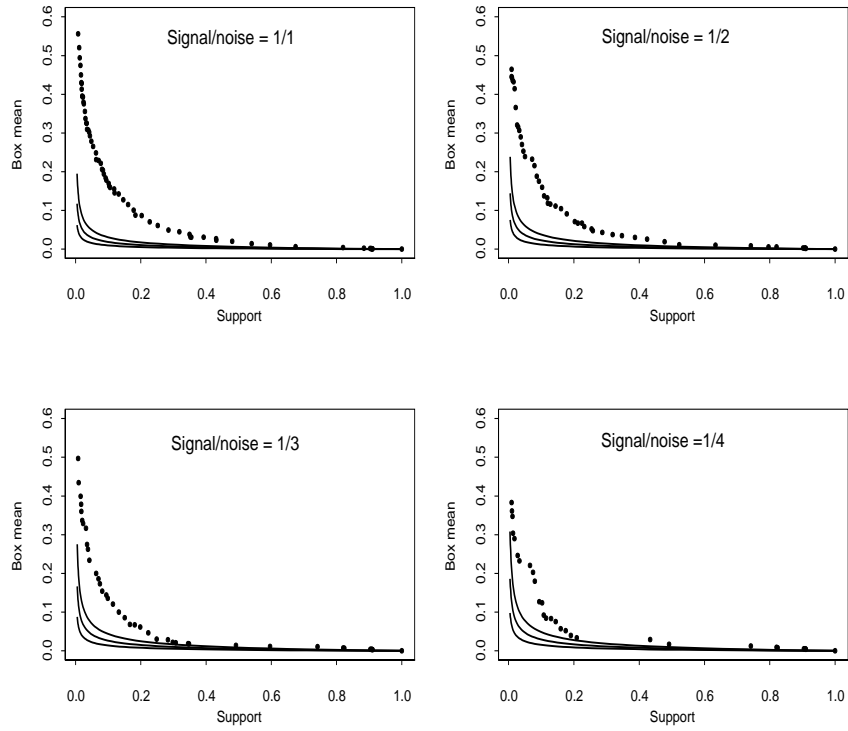


Figure 3: Undominated test data trajectory points for the simulated example of Section 19.1, with increasing amounts of noise added to the target function. The confidence interval curves assume no association between the inputs \mathbf{x} and the output y .

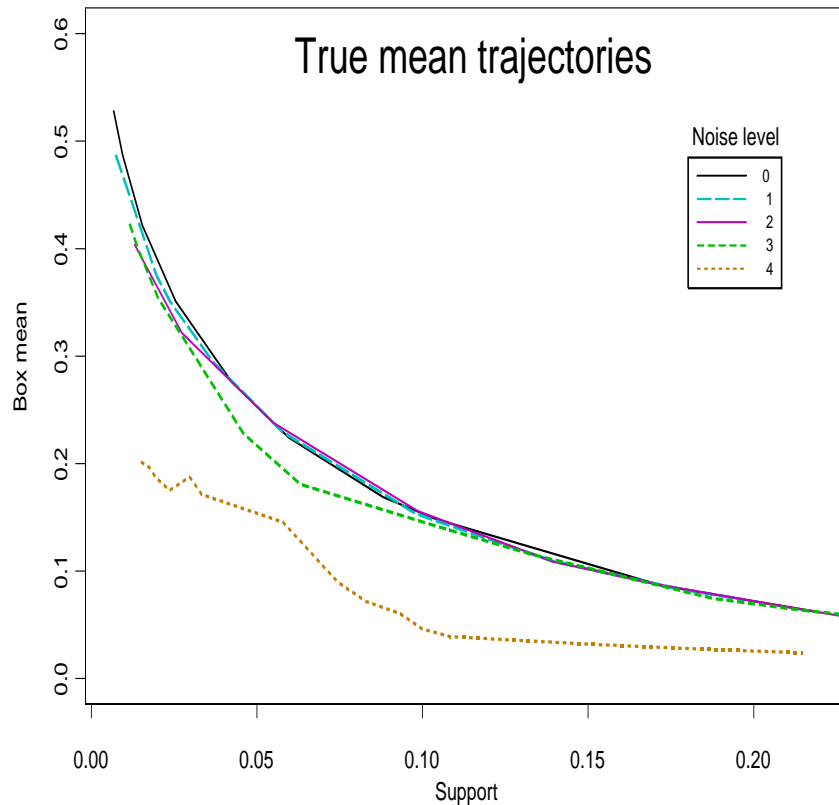


Figure 4: True mean–support values of the undominated trajectory points for the example of Section 19.1, for no output noise, and with the four noise levels shown in Fig. 3.

plots in Fig. 3. Adjacent boxes taken from the same noise level plot are connected by straight lines. Also shown for comparison are corresponding values from the no noise ($\delta = 0$) example of Section 19.1. The results for lower noise levels dominate those of higher levels, but for $\delta \in \{1, 2\}$ they are quite close to the no noise results, degrading a little at the very smallest support values ($\beta < 0.03$). For $\delta = 3$ there is substantial degradation for $\beta < 0.07$. Boxes produced at the highest noise level $\delta = 4$ are substantially inferior to the others at all support values. This is caused by the inability of the redundant variable elimination strategy in this case to unambiguously separate the first three inputs as being the only relevant ones.

These results confirm the suspicion that increasing noise on the output variable y degrades the quality of PRIM estimates, as would be the case for any estimation procedure. However, at least for this example, it appears that moderately high noise levels do not have a devastating effect on the PRIM procedure. At signal/noise ratios as low as $1/3$ it was still able to produce good quality results, and at $1/4$ perhaps useful results. These findings can only be regarded as suggestive, rather than definitive, since they were derived from only one replication of one situation. Other situations characterized by a different target function, sample size, or different \mathbf{x} distribution might be less (or more) tolerant to high levels of noise.

3 Specific Responses

In this section we respond to some of the other specific questions and criticisms raised in individual discussions. Space requirements limit us from commenting on all of the important issues and good points raised by the discussants. In particular we do not respond to points with which we for the most part agree.

Burges questions the ability of the method to perform well in the presence of many input variables ($n > 100$). The relevant question here is the nature of the target function $f(\mathbf{x}) = E[y | \mathbf{x}]$, and the intrinsic dimensionality of the distribution of the inputs \mathbf{x} . In many applications involving a large number of inputs there is a high degree of dependence among many of them. The (effective) *intrinsic* dimensionality is thereby much smaller than the actual number of inputs. For statistical estimation it is this lower intrinsic dimensionality that is relevant.

The number of relevant inputs may be further reduced by the nature of the target function. In particular, it may have a strong dependence only on a small to moderate number of them. The smaller the size of this relevant subset the more tractable the problem becomes. As noted in Section 8.2 one of the design goals of the patient induction strategy was to accommodate dependencies on a larger number of variables than is possible with greedy methods. Although this helps, it cannot overcome the basic curse-of-dimensionality. If the input variable distribution has high intrinsic dimensionality, and the true underlying target function has a strong dependence on a large number of inputs, the algorithm will have difficulty. All successful function estimation methods must make simplifying assumptions. For PRIM the assumption is that the intrinsic dimensionality and/or the number of inputs relevant to the target is moderate. Since this information is not likely to be known for any particular problem one must adopt **Titterington's** “suck-it-and-see” approach.

Feelders suggests *Data Surveyor* and **Kloesgen** suggests *Explora* as competing systems with similar goals to PRIM. The goals are indeed similar but the methodology is fairly different. Both Explora and Data Surveyor use the very greedy best first beam search strategy of the CN2 algorithm. They encapsulate the mean–support trade–off as an explicit mathematical criterion to be optimized. For example Explora uses an expression of the form $(\text{support})^a (\text{box mean} - \text{global mean})$, with $a \geq 0$ a user–specified parameter controlling the relative importance of mean and support. The nature of PRIM's patient peeling algorithm allows this to be directly assessed by the user through the peeling trajectory plot, in place of repeated runs with different values of a . Neither system seems to have an analog to the redundant variable elimination strategy of Section 10. Both systems do contain a wealth of interesting ideas from which future versions of PRIM could benefit. Hopefully the converse is also true.

Huber reports results of running his implementation of PRIM on a modified version of the simulated example of Section 19.1. The modification consist of changing the target function (19.1) to

$$y = f(\mathbf{x}) = \max_{1 \leq j \leq 3} |x_j|.$$

This function has a maximum at every point on the surface of the hypercube bounding the first three input variables. Ideally, one would like PRIM to construct six successive “solution” boxes, in six iterations of covering, each described by simple rules of the form $x_i > a_i$ or $x_i < b_i$ for $i \in \{1, 2, 3\}$. The values of $\{a_i\}_1^3$ should be $\gtrsim -1$ and those of $\{b_i\}_1^3$ should be $\lesssim 1$.

Running our implementation on this example using a single trajectory ($B = 1$) we encountered results similar to those reported by Huber. The algorithm was usually able to find the six solution boxes, but not in six iterations of covering. On five independently generated data sets, the solution boxes were found after 9 iterations of covering three times, and 10 iterations once. On one of the five replications it failed to find two of the six solution boxes after 12 iterations, at which point it ran out of data. When the number of bumped trajectories was increased to $B = 20$, the program always found all six solution boxes in the first six covering iterations. This example illustrates the utility of the multiple trajectory strategy (Section 13) to increase the power of the procedure, especially in difficult situations.

Kloesgen suggests that an interbox dissimilarity measure based on overlapping of subgroups (intersection related to union) would be more reasonable than the one (16.5) suggested in Section 16.3. Degree-of-overlap is a useful diagnostic that is provided with the current implementation of PRIM. However, it has limitations as a dissimilarity measure; all disjoint subsets have the same (zero) overlap and thus maximal dissimilarity, regardless of their locations. For example, two boxes produced by a splitting a “parent” box on one of the input variables would have maximal dissimilarity based on overlap, but zero dissimilarity based on (16.5). With (16.5) box dissimilarity increases as the subset of values on each input variable defining the two boxes increasingly differ.

Lunn takes issue with our choice of CART as a chief competitor and suggests that neural networks might be more worthy. In terms of function *approximation* neural networks often (perhaps usually) out perform CART. However, as noted by **Huber**, one is still left with the problem of finding local regions in the input space where the network approximation has large values. Since the approximating function is smooth one might try to apply numerical optimization technology, with repeated starts to uncover multiple maxima. A serious problem with this approach is constraining the search to regions of the input space supported by the data distribution. The network approximation is likely to have little validity extrapolated outside the input data distribution. One could use the convex hull of the input variable training data to define the optimizing region. However, this is an expensive computation for large samples in high dimensional spaces. Also, the actual data may only occupy a small fraction of the interior of the hull, especially if outliers are present. This must be followed by a difficult numerical optimization with linear inequality constraints.

Alternatively, one could attempt to understand the modal structure of the network approximation by investigating the subset of training data points with largest predicted output values. For example a cluster analysis might be performed on this subset. This would overcome the data support problem, but then the whole host of problems associated with cluster analysis would be introduced. Among these are the arbitrary choices of method, its procedural parameters, and the metric imposed on the input space. In addition to the general reliability of clustering algorithms, one has the problem of trying to interpret their results in the context of the geometry of the input space. This approach would substitute *two* more difficult problems, function approximation *and* cluster analysis, for the simpler one of function mode estimation.

Of course one could apply PRIM to the output of the network using the training data inputs $\{\mathbf{x}_i\}_1^N$. The network would then serve as a “presmoother” to filter out noise. This may be helpful. However, the results of the noise studies above suggest that PRIM is fairly robust against the presence of output noise, maybe in some cases more so than neural networks.

Ridgeway, Richardson, and Madigan point out that naive Bayes classifiers provide interpretable descriptions of their prediction rules. This is a consequence of the fact that they are basically additive models in the logarithms of the densities. Additive models of any kind are well known to be highly interpretable (see Hastie and Tibshirani 1989). Adding them together by boosting, bagging, or any other prescription maintains the additivity, and thus interpretability (see Friedman, Hastie and Tibshirani 1998 for interpreting boosted additive decision trees.) However, as with neural networks, one is still faced with the problem of finding regions where the model predicts large output values. If each of the additive components contain only a few modes, then one might investigate combinations of the modal values over the input variables. However, these simultaneous input values may not be within the support of the training data.

Scott takes issue with our recycling of the acronym PRIM. We are pleased that the original *PRIM9* graphical data exploration system (Fisher, Friedman and Tukey 1975) is still remembered. We thought it had long since been forgotten. As one of the present authors (JHF) was responsible for the name of that system back then, we felt we had “author’s license” to use it again in this context. Scott also points that there are some function approximators that can provide estimates of global target mean as accurate as the sample output mean. He cites the Nadaraya–Watson regression estimator. Another such method is regression trees. In fact, this is the case for any linear least–squares estimate.

Stone raises the important issue of resistance to outliers. Since PRIM uses only rank values of the input variables, there is no problem of outliers in input space. Unusually *small* values of the output variable y have little impact since they are usually removed with the first peel. However, as suggested by Stone, outlying *large* values of the output variable y can have a substantial influence on the results. The peeling algorithm will tend to produce boxes that attempt to isolate the outlying values, heavily influencing the series of boxes produced in the peeling sequence. Trajectories produced in this manner tend to have a characteristic shape; the box mean $\bar{y}_B(\beta)$ as a function of the support β is approximately $\bar{y}_B(\beta) \simeq y_o/(\beta N)$, where y_o is the outlying value. As long as this is recognized there is no problem; one can simply delete the offending outlier and resume covering. If not recognized, misleading results may be obtained.

There are a variety of methods for dealing with outlying y -values. Those that can be identified from the marginal y -distribution should be separately investigated and then deleted for further analysis. Stone suggests applying PRIM to $\tilde{y} = \log y$. Another possibility is to use the rank transform $\tilde{y} = \hat{F}(y)$, where \hat{F} is the empirical cumulative distribution of y . The most satisfying solution is to use resistant location estimates, such as trimmed means or medians, in place of means to drive the peeling. This is straight forward to implement but there will be an increased computational burden. Updating formulae for the resistant estimates are not nearly as fast as that for the mean.

Again we thank the contributors for a lively discussion and emphasize that space constraints have limited our response to only a small fraction of the helpful suggestions and important criticisms raised by the discussants.

References

- [1] Fisher-Keller, M. A., Friedman, J. H. and Tukey, J. W. (1975). *Prim9*: a data display and analysis system. Proceedings: Pacific Regional Conference of the Association for Computing Machinery, San Francisco, CA.
- [2] Friedman, J. H. (1998). *SuperGEM*.
<http://www-stat.stanford.edu/~jhf/SuperGEM>.
- [3] Friedman, J. H., Hastie, T. and Tibshirani, R. (1998). Additive logistic regression: a statistical view of boosting. Dept. of Statistics, Stanford University Technical Report.
- [4] Hastie, T. and Tibshirani, R. (1989). *Generalized Additive Models*. Chapman & Hall.