The Annals of Statistics
2004, Vol. 32, No. 1, 233–244
© Institute of Mathematical Statistics, 2004

ON THE FORWARD AND BACKWARD ALGORITHMS OF PROJECTION PURSUIT[1]

BY MU ZHU

University of Waterloo

This article provides a historic review of the forward and backward projection pursuit algorithms, previously thought to be equivalent, and points out an important difference between the two. In doing so, a small error in the original exploratory projection pursuit article by Friedman [*J. Amer. Statist. Assoc.* **82** (1987) 249–266] is corrected. The implication of the difference is briefly discussed in the context of an application in which projection pursuit density estimation is used as a building block for nonparametric discriminant analysis.

1. Introduction. Let $p(\mathbf{x})$ be an arbitrary high-dimensional density function. Estimating $p(\mathbf{x})$ from data $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathbb{R}^d$ is a difficult problem. Projection pursuit density estimation [Friedman, Stuetzle and Schroeder (1984) and Friedman (1987)] is one of the few relatively successful methods. Two algorithms exist—one originally proposed by Friedman, Stuetzle and Schroeder (1984), which we refer to as the forward algorithm, and one subsequently proposed by Friedman (1987), which we refer to as the backward algorithm. The two algorithms have always been regarded as equivalent. In this article, we show that the two algorithms are only equivalent in the context of projection pursuit density approximation [Huber (1985)]; in the context of projection pursuit density estimation, however, the two algorithms will be shown to have substantial differences.

2. Projection pursuit density approximation. In projection pursuit density approximation [Huber (1985)], we assume that $p(\mathbf{x})$ is known and wish to construct an approximation for it with

$$p_M(\mathbf{x}) = p_0(\mathbf{x}) \prod_{m=1}^{M} f_m(\boldsymbol{\alpha}_m^T \mathbf{x}), \tag{1}$$

where $\|\boldsymbol{\alpha}_m\| = 1$ is a unit vector.

2.1. *Forward approximation.* In the forward approximation (called the synthetic approach by Huber), we start with an initial approximation $p_0(\mathbf{x})$, often a simple parametric model such as a multivariate Gaussian, and find a series of ridge modifications to get closer to the actual density function itself. This can be done

Received April 2002; revised May 2003.

[1]Supported in part by the Natural Science and Engineering Research Council of Canada.

AMS 2000 subject classification. 62H40.

Key words and phrases. Density estimation, nonparametric discriminant analysis.

recursively: at the mth iteration, let $p_{m-1}(\mathbf{x})$ be the latest approximation of $p(\mathbf{x})$; the goal, then, is to find $\boldsymbol{\alpha} \in \mathbb{R}^d$ and $f_m(\cdot)$ to update the approximation by

$$p_m(\mathbf{x}) = p_{m-1}(\mathbf{x}) f_m(\boldsymbol{\alpha}^T \mathbf{x}).$$

It can be shown [Huber (1985) and Friedman, Stuetzle and Schroeder (1984)] that for every fixed $\boldsymbol{\alpha}$, the optimal ridge modification $f_m(\boldsymbol{\alpha}^T\mathbf{x})$ that makes $p_m(\mathbf{x})$ the closest to $p(\mathbf{x})$ (in terms of cross-entropy; see below) is given by

$$f_m(\boldsymbol{\alpha}^T\mathbf{x}) = \frac{p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}{p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}. \tag{2}$$

This is quite intuitive: for a given direction $\boldsymbol{\alpha}$, we simply divide out the marginal of the current approximation and replace it with the correct marginal of the true density. If we keep on doing this in all projections, we eventually get the true density itself. In practice we update only in a few directions to get the best approximation. This is achieved by choosing $\boldsymbol{\alpha}$ to maximize the cross-entropy between $p_m(\mathbf{x})$ and $p(\mathbf{x})$ at every step, that is,

$$\max_{\boldsymbol{\alpha}} \int \log p_m(\mathbf{x}) p(\mathbf{x})\, d\mathbf{x} \quad \text{s.t.} \quad \int p_m(\mathbf{x})\, d\mathbf{x} = 1. \tag{3}$$

Summarizing all this together (also see Algorithm 1), we arrive at the forward approximation of $p(\mathbf{x})$:

$$p_0(\mathbf{x}) \prod_{m=1}^{M} \frac{p^{(\alpha_m)}(\boldsymbol{\alpha}_m^T\mathbf{x})}{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T\mathbf{x})}.$$

ALGORITHM 1 (Forward projection pursuit density approximation).

- Start with an initial approximation $p_0(\mathbf{x})$, say, $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.
- For $m = 1$ to M:

  1. Choose a direction, $\boldsymbol{\alpha}$, in which the distance between $p_{m-1}(\mathbf{x})$ and $p(\mathbf{x})$ is the largest. Here distance can be measured, for example, by the cross-entropy.
  2. Calculate $p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ and $p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$.
  3. Update the approximation by

  $$p_m(\mathbf{x}) = p_{m-1}(\mathbf{x}) \frac{p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}{p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}.$$

  End For.

2.2. *Backward approximation.* Instead of constructing the approximation in a forward fashion from an initial $p_0(\mathbf{x})$, Huber (1985) pointed out that the whole process can be turned backwards (called the analytic approach by Huber). That is, start with $p_0(\mathbf{x}) = p(\mathbf{x})$ and recursively modify it toward a target density $q(\mathbf{x})$

by a series of ridge modifications, where $q(\mathbf{x})$ is a simple known density, say, the standard Gaussian. The entire process (see Algorithm 2) leads to the model

$$p_M(\mathbf{x}) = p_0(\mathbf{x}) \prod_{m=1}^{M} \frac{q^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})}{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})},$$

where $p_M(\mathbf{x})$ is sufficiently close to $q(\mathbf{x})$. Turning this equation backward and replacing $q(\mathbf{x})$ with $p_M(\mathbf{x})$, we can then approximate the original density $p(\mathbf{x}) = p_0(\mathbf{x})$ with

$$q(\mathbf{x}) \prod_{m=1}^{M} \frac{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})}{q^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})}. \tag{4}$$

Clearly, there is no apparent difference between the forward approach and the backward approach in this context. We shall see that this is no longer true for density estimation.

ALGORITHM 2 (Backward projection pursuit density approximation).

- Select a target density $q(\mathbf{x})$. It is often convenient to use a simple density here, for example, the standard multivariate Gaussian.
- Start with $p_0(\mathbf{x}) = p(\mathbf{x})$, the actual density we wish to approximate.
- For $m = 1$ to M:

  1. Choose a direction, $\boldsymbol{\alpha}$, for ridge modification. Usually, we choose a direction where the distance between $p_{m-1}(\mathbf{x})$ and $q(\mathbf{x})$ is the largest. Again, the distance can be measured by cross-entropy.
  2. Modify $p_{m-1}(\mathbf{x})$ in the $\boldsymbol{\alpha}$ direction toward $q(\mathbf{x})$. After the modification, the new density becomes

  $$p_m(\mathbf{x}) = p_{m-1}(\mathbf{x}) \frac{q^{(\alpha)}(\boldsymbol{\alpha}^T \mathbf{x})}{p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T \mathbf{x})}.$$

  End For.
- Assuming $p_M(\mathbf{x}) \approx q(\mathbf{x})$, approximate $p(\mathbf{x})$ with

$$q(\mathbf{x}) \prod_{m=1}^{M} \frac{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})}{q^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \mathbf{x})}.$$

3. Projection pursuit density estimation. Density estimation is a more practical problem than density approximation because, in practice, the density function $p(\mathbf{x})$ is often unknown; it must be estimated from data. We can adapt the forward and backward projection pursuit density approximation techniques outlined above for projection pursuit density estimation.

3.1. *Adapting the forward approach.* It is conceptually easy to adapt the forward density approximation procedure for density estimation. Here, we start with an initial density estimate $p_0(\mathbf{x})$ using a parametric model such as the multivariate Gaussian. Then recursively, given the current estimate $p_{m-1}(\mathbf{x})$, we must estimate the ridge modification $f_m(\boldsymbol{\alpha}^T\mathbf{x})$. From (2), it is clear that we need to estimate two univariate density functions $p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ and $p^{(\alpha)}_{m-1}(\boldsymbol{\alpha}^T\mathbf{x})$. In fact, for the purpose of finding the best ridge direction, $\boldsymbol{\alpha}$, we need to estimate at least their first derivatives as well. This is because the maximization problem (3) must be solved numerically. Also note that since $p(\mathbf{x})$ is unknown, we must replace

$$\int \log p_m(\mathbf{x})p(\mathbf{x})\,d\mathbf{x}$$

with an empirical estimate

$$\frac{1}{n}\sum_{i=1}^{n}\log p_m(\mathbf{x}_i) \qquad \text{where } \mathbf{x}_i \sim p(\mathbf{x}).$$

For the current discussion, however, we treat the estimation of these univariate density functions and their derivatives as a single block, but keep in mind that it involves derivative estimation as well.

For a fixed direction, $\boldsymbol{\alpha}$, $p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ is easy to estimate: we simply project the data onto $\boldsymbol{\alpha}$ to get $z_i = \boldsymbol{\alpha}^T\mathbf{x}_i$ and then estimate a univariate density (and its derivative) from z_i. The difficulty lies in the estimation of $p^{(\alpha)}_{m-1}(\boldsymbol{\alpha}^T\mathbf{x})$, the marginal density of the current density estimate (and the related derivative). The solution proposed in Friedman, Stuetzle and Schroeder (1984) is to use Monte Carlo sampling:

1. Since we can evaluate the current estimate, $p_{m-1}(\mathbf{x})$, we can use the Monte Carlo method to draw a sample from $p_{m-1}(\mathbf{x})$, $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$.
2. Given $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\} \sim p_{m-1}(\mathbf{x})$, $p^{(\alpha)}_{m-1}(\boldsymbol{\alpha}^T\mathbf{x})$ is then estimated by standard techniques using the projected data, $\{\boldsymbol{\alpha}^T\mathbf{u}_1, \boldsymbol{\alpha}^T\mathbf{u}_2, \dots, \boldsymbol{\alpha}^T\mathbf{u}_N\}$.

Here we omit the details of Monte Carlo sampling. It is a separate topic and is not crucial for the present discussion; a clear outline is given in the Appendix of Friedman, Stuetzle and Schroeder (1984).

3.2. *Adapting the backward approach.* The need to incorporate a Monte Carlo step makes the computation of the forward algorithm quite cumbersome. Alternatively, we may wish to adapt the backward density approximation procedure, but it is not difficult to see from the discussion above (also see Algorithm 2) that a difficulty remains similar to that in the forward algorithm: we must estimate $p^{(\alpha_m)}_{m-1}(\cdot)$, the marginal density of $p_{m-1}(\mathbf{x})$, at every step. Actually, this is even harder to achieve in this case. Because we start with something we know in the forward algorithm, we can actually evaluate $p_{m-1}(\mathbf{x})$ at every iteration; this allows us to draw Monte Carlo samples from it. In the backward algorithm, we start with the density we wish to estimate, so we do not even know how to evaluate $p_{m-1}(\mathbf{x})$ in general. As a result, even the computationally expensive Monte Carlo step is infeasible here.

4. Exploratory projection pursuit. A feasible backward projection pursuit density estimation algorithm later appeared in the form of exploratory projection pursuit [Friedman (1987)]. The key idea is as follows: at every iteration the density function changes due to the ridge modification, so we transform the data as we go along so as to conform to the evolving density. The algorithm is given here as Algorithm 3. The data transformation step within every iteration is the key element that makes the backward algorithm work, because we can now estimate $p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ directly from the (transformed) data. In fact, we do not even need to resort to the Monte Carlo method. In this sense, this algorithm provides a significant improvement over the forward algorithm. However, as we show in the next section, a price has to be paid for this computational simplification. This algorithm actually changes the density model in (4). Friedman (1987) did not realize this difference; nor has this difference been discovered elsewhere. Due to this difference, we argue that the forward and backward algorithms for projection pursuit are *not* equivalent at the practical level.

ALGORITHM 3 (Exploratory projection pursuit).

- Initially, $\mathbf{x}_i$ follows density function $p_0(\mathbf{x}) = p(\mathbf{x})$, the actual density we wish to estimate.
- For $m = 1$ to M:

  1. Choose a direction, $\boldsymbol{\alpha}$, for ridge modification, as in Algorithm 2.
  2. Construct a transformation $\mathbf{x}' = h(\mathbf{x})$ so that the marginal density of $\mathbf{x}'$ agrees with the target density, $q(\cdot)$, in the $\boldsymbol{\alpha}$ direction, while in all directions orthogonal to $\boldsymbol{\alpha}$ it agrees with $p_{m-1}(\cdot)$. For convenience, q is often chosen to be the standard Gaussian density function. Then, for any direction $\boldsymbol{\alpha}$, $q^{(\alpha)}(\cdot) = g(\cdot)$ is the standard univariate Gaussian density. The transformation $h(\cdot)$ can be constructed as

     $$h(\mathbf{x}) = \mathbf{A}^{-1}(t(\mathbf{A}\mathbf{x})),$$

     where $\mathbf{A}$ is an orthogonal rotation matrix such that

     $$\mathbf{z} \stackrel{\text{def}}{=} \mathbf{A}\mathbf{x} = \begin{pmatrix} \boldsymbol{\alpha}^T\mathbf{x} \\ \mathbf{A}^*\mathbf{x} \end{pmatrix}; \tag{5}$$

     such $\mathbf{A}$ can be constructed using the Gram–Schmidt procedure, [e.g., Friedberg, Insel and Spence (1989), page 307] and the transformation t is given by

     $$t(z_j) = \begin{cases} \gamma(z_j), & \text{for } j = 1, \\ z_j, & \text{for } j > 1. \end{cases} \tag{6}$$

Here, $\gamma(\cdot)$ is a monotonic transformation such that

$$\gamma(z_1) \overset{\text{def}}{=} \Phi^{-1}\big(F_{z_1}(z_1)\big), \tag{7}$$

where $\Phi(\cdot)$ is the cumulative distribution function (c.d.f.) of the standard normal and F_{z_1} is the marginal c.d.f. of $z_1 = \boldsymbol{\alpha}^T\mathbf{x}$, which can be estimated by the empirical distribution function. By definition, $\gamma(z_1)$ follows the standard Gaussian distribution. The entire transformation $h(\mathbf{x})$ can be represented as

$$\begin{tikzcd} \mathbf{x} \arrow[r, "h"] \arrow[d, "\mathbf{A}"] & h(\mathbf{x}) \\ \mathbf{z} \arrow[r, "t"] & t(\mathbf{z}). \arrow[u, "\mathbf{A}^{-1}"'] \end{tikzcd} \tag{8}$$

In the next section we show that the transformed data, $h(\mathbf{x})$, has density

$$p_m(h(\mathbf{x})) = p_{m-1}(\mathbf{x})\frac{g(\gamma(\boldsymbol{\alpha}^T\mathbf{x}))}{p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}. \tag{9}$$

3. Let

$$\mathbf{x}_i \leftarrow h(\mathbf{x}_i),$$

so that after the transformation $\mathbf{x}_i$ follows density function $p_m(\cdot)$.

End For.

5. Nonequivalence.

We are now ready to study the exploratory projection pursuit algorithm (Algorithm 3) in more detail. First, we show that (9) is true.

LEMMA 1. *Suppose $p(x, \mathbf{y})$ is the density function for $(x, \mathbf{y})$, where $x \in \mathbb{R}$ and $\mathbf{y} \in \mathbb{R}^{d-1}$. Let*

$$x' = \Phi^{-1}\big(F_x(x)\big) \overset{def}{=} \gamma(x), \tag{10}$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal and F_x the marginal c.d.f. of x. Let $p'(x', \mathbf{y})$ be the density for $(x', \mathbf{y})$. Then

$$p'(x', \mathbf{y}) = p\big(\gamma^{-1}(x'), \mathbf{y}\big)\left(\frac{g(x')}{p_x(\gamma^{-1}(x'))}\right). \tag{11}$$

REMARK. While (11) is the standard way to represent the density for $(x', \mathbf{y})$, we prefer to write it as

$$p'\big(\gamma(x), \mathbf{y}\big) = p(x, \mathbf{y})\left(\frac{g(\gamma(x))}{p_x(x)}\right), \tag{12}$$

because $(x, \mathbf{y})$ is the original variable, and a transformation is applied to obtain $(x', \mathbf{y}) = (\gamma(x), \mathbf{y})$. Equation (12) allows us to keep track of the original and the transformed data more directly. This comes in handy as we go a few steps into the recursive iteration, where it is much easier to keep track of a series of γ's rather than a series of γ^{-1}'s.

THEOREM 1. *Let $p(\mathbf{x})$ be the density function for $\mathbf{x}$. Then the density function for $h(\mathbf{x})$, where $h(\mathbf{x})$ is defined in* (8), *is given by*

$$p'(h(\mathbf{x})) = p(\mathbf{x})\left(\frac{g(\gamma(\boldsymbol{\alpha}^T\mathbf{x}))}{p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}\right), \tag{13}$$

where $p^{(\alpha)}(\cdot)$ is the marginal density of $\mathbf{x}$ in the $\boldsymbol{\alpha}$ direction.

The proofs for the lemma and the theorem are given in the Appendix. The validity of (9) follows directly from Theorem 1. With this result, we can now work out the details of the exploratory projection pursuit algorithm and see that it transforms the density in successive steps, according to

$$\begin{aligned}
p_0(\mathbf{x}) &= p(\mathbf{x}),\\
p_1(h_1(\mathbf{x})) &= p_0(\mathbf{x})\left(\frac{g(\gamma_1(\boldsymbol{\alpha}_1^T\mathbf{x}))}{p_0^{(\alpha_1)}(\boldsymbol{\alpha}_1^T\mathbf{x})}\right),\\
p_2(h_2(h_1(\mathbf{x}))) &= p_1(h_1(\mathbf{x}))\left(\frac{g(\gamma_2(\boldsymbol{\alpha}_2^T h_1(\mathbf{x})))}{p_1^{(\alpha_2)}(\boldsymbol{\alpha}_2^T h_1(\mathbf{x}))}\right)
\end{aligned}$$

and so on, where $h_m(\mathbf{x})$ transforms $\mathbf{x}$ in the $\boldsymbol{\alpha}_m$ direction as defined by (8), leaving all orthogonal directions (relative to $\boldsymbol{\alpha}_m$) unchanged. To simplify the notation, write

$$\begin{aligned}
\rho_0(\mathbf{x}) &= \mathbf{x},\\
\rho_1(\mathbf{x}) &= h_1(\mathbf{x}),\\
\rho_2(\mathbf{x}) &= h_2 \cdot h_1(\mathbf{x}),\\
&\vdots\\
\rho_M(\mathbf{x}) &= h_M \cdot h_{M-1} \cdots h_1(\mathbf{x}),
\end{aligned}$$

and we have, at the end of M iterations,

$$p_M(\rho_M(\mathbf{x})) = p_0(\mathbf{x}) \prod_{m=1}^{M} \left(\frac{g(\gamma_m(\boldsymbol{\alpha}_m^T \rho_{m-1}(\mathbf{x})))}{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \rho_{m-1}(\mathbf{x}))}\right), \tag{14}$$

where, again, $p_M(\cdot)$ is sufficiently close to $q(\cdot)$. Turning this backwards and replacing $q(\cdot)$ for $p_M(\cdot)$, we obtain a model for estimating the original density $p(\mathbf{x})$:

$$q(\rho_M(\mathbf{x})) \prod_{m=1}^{M} \left(\frac{p_{m-1}^{(\alpha_m)}(\boldsymbol{\alpha}_m^T \rho_{m-1}(\mathbf{x}))}{g(\gamma_m(\boldsymbol{\alpha}_m^T \rho_{m-1}(\mathbf{x})))} \right). \tag{15}$$

Note that this is *not* the same as (4). The resulting expression of the estimate is much more cumbersome; it does not unwrap as nicely as in the forward algorithm.

Why is there such a big difference? In the forward algorithm, no transformation is introduced to the original data and we must estimate $p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ from a Monte Carlo sample, but in the backward algorithm, we actually *transform* the data at every step. This eliminates the need for Monte Carlo sampling, because the transformation makes the data conform to the density $p_m(\mathbf{x})$ rather than to the original density $p_0(\mathbf{x}) = p(\mathbf{x})$. This allows us to estimate $p_{m-1}^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})$ directly from the (transformed) data. In fact, recall that if we did not introduce these transformations, the backward algorithm would have been hopeless in practice, because even Monte Carlo sampling would have been infeasible (see Section 3.2). However, to wrap around the entire process to obtain an estimate of the *original* density (prior to any of the transformations induced within the algorithm), we must unfold the entire sequence of transformations as well.

In the original exploratory projection pursuit article [Friedman (1987)], this important detail was overlooked and the resulting model (15) from the exploratory projection pursuit algorithm was mistakenly thought to be the same as (4). The focus of the Friedman's (1987) article was to introduce the data transformation step, which, as we have argued above, is crucial for making the backward algorithm work. Because things work out nicely in the forward algorithm and because the two algorithms are equivalent in the context of density approximation, it is easy to believe that the backward algorithm must work in the same way, without paying close attention to the fact that the added transformation would have made the model much more complex.

Here is a summary of this discussion: in the forward algorithm, we must pay a price in Monte Carlo sampling; in the backward algorithm, although we can avoid the Monte Carlo step, we must pay a price (not so much a computational one) to keep track of the sequence of transformations. Either way, nothing comes out entirely free. Despite the computational savings, the resulting density model from the backward algorithm is much more cumbersome, because it involves the entire sequence of transformations—although theoretically this model can still be used for density estimation as long as good track is kept of all the transformations. Notice that all of these transformations involve only an orthogonal rotation and a one-dimensional monotonic transform. In terms of computational cost, this is indeed cheaper than Monte Carlo methods. However, the resulting model *is* cumbersome. In fact, it is no longer a clean-cut projection pursuit density model, because the transformation $\rho_{m-1}(\mathbf{x})$ is nested inside the ridge function f_m.

6. Implication. The nonequivalence between the forward and backward algorithms has some important practical implications if you wish to use projection pursuit density estimation as a building block. For example, in discriminant analysis, you may wish to model the density function in each class nonparametrically using projection pursuit density estimation. In particular, you can model the density for class k, $k = 1, 2, \dots, K$, as

$$p_k(\mathbf{x}) = p_0(\mathbf{x}) \prod_{m=1}^{M} f_{mk}(\boldsymbol{\alpha}_m^T \mathbf{x}). \tag{16}$$

That is, start with a *common* initial guess for all the classes and recursively augment each density with a series of ridge modifications to distinguish the classes. These ridge directions are often called *discriminant directions*. Given a direction $\boldsymbol{\alpha}$, the best ridge modification f_{mk} is the same as (2) for each class k. However, $\boldsymbol{\alpha}$ would have to be chosen differently. Instead of choosing $\boldsymbol{\alpha}$ to minimize the distance between the estimated and the true density functions, choose $\boldsymbol{\alpha}$ to be a discriminant direction, that is, the direction that separates the classes as much as possible. Class separation can be measured by the likelihood–ratio statistic [e.g., Zhu and Hastie (2003)] or by the misclassification rate [e.g., Polzehl (1995)].

There are several immediate advantages to this approach. For example, the complexity of the model can be easily *regularized* by selecting only a few directions where there are significant differences between classes. Therefore, although this is a density-based classification method, it actually does *not* waste any effort in directions which may contain interesting features in the density function itself but do not contribute to class differentiation. The most attractive aspect of such an approach is probably the following: because the initial density is *common* for all the classes, the decision boundary between class k and K depends *only* on the ratio of the augmenting ridge functions. In particular, the decision boundary is characterized by

$$\prod_{m=1}^{M} \frac{f_{mk}(\boldsymbol{\alpha}_m^T \mathbf{x})}{f_{mK}(\boldsymbol{\alpha}_m^T \mathbf{x})} \overset{\text{def}}{=} \prod_{m=1}^{M} g_{mk}(\boldsymbol{\alpha}_m^T \mathbf{x}) = 1$$

or

$$\sum_{m=1}^{M} \log\bigl(g_{mk}(\boldsymbol{\alpha}_m^T \mathbf{x})\bigr) = 0,$$

which makes classification very easy.

However, it is not hard to see that if the density estimation step is implemented using the backward projection pursuit algorithm, then things become rather complicated. According to (14), the final density model for class k would become

$$p_k(\mathbf{x}) = p_M\bigl(\rho_{M,k}(\mathbf{x})\bigr) \prod_{m=1}^{M} \frac{1}{f_{mk}(\boldsymbol{\alpha}_m^T \rho_{m-1,k}(\mathbf{x}))}. \tag{17}$$

This gives rise to several important differences:

- Although $p_M(\cdot)$ can be regarded as the same for all classes, the $\rho_{M,k}(\mathbf{x})$ inside it is not the same for all k. Therefore, we still need $p_M(\cdot)$ for classification purposes. Thus, unlike in the forward algorithm, the decision boundary between any two classes will no longer just depend on the ratio of a series of ridge functions.
- In fact, because the data are transformed differently at each iteration for each class, the ratio of two resulting "ridge functions" (for classes k and K) becomes

$$\frac{f_{mk}(\boldsymbol{\alpha}_m^T \rho_{m-1,k}(\mathbf{x}))}{f_{mK}(\boldsymbol{\alpha}_m^T \rho_{m-1,K}(\mathbf{x}))}.$$

  This is no longer a simple ridge function due to the function $\rho_{m-1,k}(\cdot)$, which is different for each class k.
- That we need p_M for classification is a serious problem, since p_M is still a full multidimensional density function which we must estimate. This is because in this application, we do not usually expect p_M to be close enough to the target density q. Since the goal is discrimination rather than density estimation itself, it will often be the case that we have only changed the original density in a few discriminant directions. Therefore, even though p_M is an adequate density function for $\rho_{M,k}(\mathbf{x})$ regardless of k, it is likely that p_M is still quite different from q.

Of course, these differences do not make discriminant analysis impossible, but it is quite clear that a lot of the nice features of using projection pursuit density estimation for nonparametric discriminant analysis—such as the ease to construct the decision boundary—are lost in this cumbersome model.

7. Conclusion. We have pointed out an important difference between the forward and backward projection pursuit algorithms and illustrated that this difference is practically significant. Because of this difference, every user of projection pursuit must now face a trade-off between computational complexity and model tractability. It is a decision that is best left for the users to decide in the context of their specific problems.

APPENDIX

PROOF OF THEOREM 1. Recall that $h(\mathbf{x})$ is defined as

$$\begin{tikzcd} \mathbf{x} \arrow[r, "h"] \arrow[d, "\mathbf{A}"] & h(\mathbf{x}) \\ \mathbf{z} \arrow[r, "t"] & t(\mathbf{z}), \arrow[u, "\mathbf{A}^{-1}"'] \end{tikzcd}$$

where $\mathbf{A}$ and t are defined in (5) and (6), respectively. The following diagram summarizes my notation for the density functions of $\mathbf{x}$, $\mathbf{z}$, $h(\mathbf{x})$ and $t(\mathbf{z})$:

$$\begin{tikzcd}
p(\mathbf{x}) \arrow[r, "h"] \arrow[d, "\mathbf{A}"] & P'(h(\mathbf{x})) \\
p_z(\mathbf{z}) \arrow[r, "t"] & p'_z(t(\mathbf{z})). \arrow[u, "\mathbf{A}^{-1}"']
\end{tikzcd}$$

It now follows from the lemma that in the $\mathbf{z}$ space,

$$p'_z(t(\mathbf{z})) = p_z(\mathbf{z})\left(\frac{g(\gamma(z_1))}{p_{z_1}(z_1)}\right), \tag{18}$$

where $p_{z_1}(\cdot)$ is the marginal density of z_1, but the fact that $\mathbf{z} = \mathbf{A}\mathbf{x}$ implies that

$$p_z(\mathbf{z}) = \frac{1}{|\mathbf{A}|}p(\mathbf{A}^{-1}\mathbf{z}) = \frac{1}{|\mathbf{A}|}p(\mathbf{x})$$

and

$$p'_z(t(\mathbf{z})) = \frac{1}{|\mathbf{A}|}p'\big(\mathbf{A}^{-1}t(\mathbf{z})\big) = \frac{1}{|\mathbf{A}|}p'(h(\mathbf{x})).$$

Hence, (18) now implies

$$p'(h(\mathbf{x})) = p(\mathbf{x})\left(\frac{g(\gamma(\boldsymbol{\alpha}^T\mathbf{x}))}{p^{(\alpha)}(\boldsymbol{\alpha}^T\mathbf{x})}\right),$$

since by the definition of $\mathbf{A}$, we have $z_1 = \boldsymbol{\alpha}^T\mathbf{x}$. $\square$

PROOF OF LEMMA 1. By conditioning, we have

$$p'(x', \mathbf{y}) = p'(\mathbf{y}|x')p'_{x'}(x').$$

Since we applied only a monotonic transformation to x and did nothing to $\mathbf{y}$, we have, for $x' = \gamma(x)$,

$$p'(\mathbf{y}|x') = p(\mathbf{y}|x),$$

and by definition of x',

$$p'_{x'}(x') = g(x'),$$

where $g(\cdot)$ is the standard normal density. Hence,

$$\begin{aligned}
p'(x', \mathbf{y}) &= p(\mathbf{y}|x)g(x') \\
&= \left(\frac{p(x, \mathbf{y})}{p_x(x)}\right)g(x') \\
&= p(\gamma^{-1}(x'), \mathbf{y})\left(\frac{g(x')}{p_x(\gamma^{-1}(x'))}\right)
\end{aligned}$$

or

$$p'(\gamma(x), \mathbf{y}) = p(x, \mathbf{y})\left(\frac{g(\gamma(x))}{p_x(x)}\right)$$

because $x' = \gamma(x)$ and $x = \gamma^{-1}(x')$. $\square$

Acknowledgments. I thank Professor Trevor Hastie, Professor Robert Tibshirani and Professor Jerome Friedman for their suggestions and encouragement, as well as the Editor, Professor John I. Marden, an Associate Editor and two referees for their insights and helpful comments.

REFERENCES

FRIEDBERG, S. H., INSEL, A. J. and SPENCE, L. E. (1989). *Linear Algebra*, 2nd ed. Prentice-Hall, Englewood Cliffs, NJ.

FRIEDMAN, J. H. (1987). Exploratory projection pursuit. *J. Amer. Statist. Assoc.* **82** 249–266.

FRIEDMAN, J. H., STUETZLE, W. and SCHROEDER, A. (1984). Projection pursuit density estimation. *J. Amer. Statist. Assoc.* **79** 599–608.

HUBER, P. J. (1985). Projection pursuit (with discussion). *Ann. Statist.* **13** 435–525.

POLZEHL, J. (1995). Projection pursuit discriminant analysis. *Comput. Statist. Data Anal.* **20** 141–157.

ZHU, M. and HASTIE, T. J. (2003). Feature extraction for nonparametric discriminant analysis. *J. Comput. Graph. Statist.* **12** 101–120.

DEPARTMENT OF STATISTICS
AND ACTUARIAL SCIENCE
UNIVERSITY OF WATERLOO
WATERLOO, ONTARIO
CANADA
N2L 3G1
E-MAIL: m3zhu@uwaterloo.ca