Chapter 12
Earth Mover's Distance-Based Local Discriminant Basis

Bradley Marchand and Naoki Saito

Abstract Local discriminant basis (LDB) is a tool to extract useful features for signal and image classification problems. Original LDB methods rely on the time–frequency energy distribution of classes or empirical probability densities, with some information theoretic measure (such as Kullback–Leibler divergence) for feature selection. Depending on the problem, energy distributions may not provide the best information for classification. Further, training set sizes and accuracy in the computed empirical probability density functions (epdfs) may hinder the learning process. To improve these deficiencies and provide a more data adaptive algorithm, we propose the use of signatures and earth mover's distance (EMD). Signatures and EMD provide a data adaptive statistic that is more descriptive than the distribution of energies and more robust than an epdf-based approach. In this chapter, we first review LDB and EMD and then outline how they can be incorporated into a fast EMD based LDB algorithm.We then demonstrate the capabilities of our new algorithm in comparison to both energy distribution and epdf-based LDB algorithms using four different classification problems using synthetic datasets.

12.1 Introduction

Local discriminant basis (LDB) is a best basis algorithm developed by Saito and Coifman for the purpose of classification [9, 10]. It works by decomposing training signals into a time–frequency dictionary, such as block discrete cosine transform, local cosine transform, or wavelet packet transform (WPT). The dictionaries

B. Marchand (✉)
Panama City Division, Naval Surface Warfare Center, FL 32407, USA
e-mail: bradley.marchand@navy.mil

N. Saito
Department of Mathematics, University of California, Davis, CA 95616, USA
e-mail: saito@math.ucdavis.edu

X. Shen and A.I. Zayed (eds.), *Multiscale Signal Analysis and Modeling*,
DOI 10.1007/978-1-4614-4145-8_12,

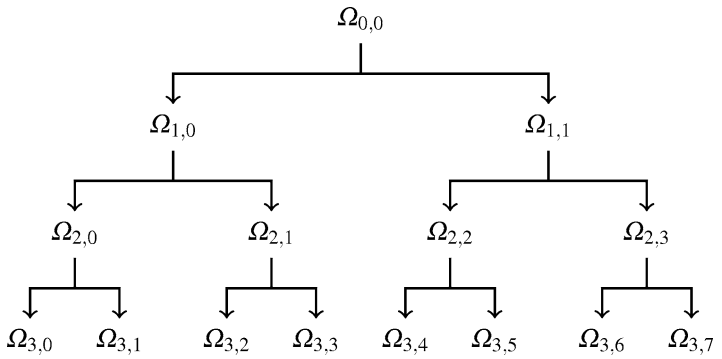

Fig. 12.1 Depiction of the wavelet packet transform. The process is the same as the wavelet decomposition with the added decomposition of the high-pass coefficients. This yields a redundant decomposition

decompose signals into a redundant set of orthogonal subspaces, as shown in Fig. 12.1. Each subspace contains basis vectors localized in time and frequency. Its goal, given a dictionary, is to find the signal representation within the dictionary that is most useful for classification and discrimination. The idea is that these dictionaries provide us with localized elementary building blocks for isolating critical differences among signal classes. These differences are learned from LDB's time–frequency map. In the original LDB algorithm, this map was a simple accumulation of class signal energy at each coordinate in each subspace. Formally, let N_c be the number of signals belonging to class c, $\left\{\mathbf{x}_i^{(c)}\right\}_{i=1}^{N_c}$ be the set of signals belonging to class c, and $\mathbf{w}_{j,k,l}$ be basis vectors from our selected dictionary parameterized by indices j, k, and l indicating the scale (or level of decomposition), frequency band, and position of basis vector, respectively. Then, our energy map for class c is formed as,

$$\Gamma_{j,k,l}^{(c)} := \frac{\sum_{i=1}^{N_c}\left(\mathbf{w}_{j,k,l}\cdot\mathbf{x}_i^{(c)}\right)^2}{\sum_{i=1}^{N_c}\left\|\mathbf{x}_i^{(c)}\right\|^2}.$$

Later Saito et al. [11] proposed a refinement of the algorithm by changing the time–frequency map from an accumulation of class signal energy to statistical distributions of the expansion coefficients. Although any distribution metric can be used to evaluate the discriminating power of a coordinate, average shifted histograms (ASH) [12] were used to compute an empirical probability density function (epdf) in [11] for their computational efficiency. This improvement allows LDB to detect finer differences because the statistical behavior of the class signals in each coordinate in the dictionary can be analyzed.

In this chapter, to further refine LDB's time–frequency map, we propose the use of *signatures* instead of epdfs and the use of *earth mover's distance* (EMD) [7] to compute the discriminating power of a coordinate. Signatures provide us with a fully data-driven representation, which can be efficiently used with EMD. This representation is more efficient than a histogram and is able to represent complex data structure with fewer samples. EMD is a metric between signatures that naturally extends the notion of distance between points to that of sets or distributions of elements. We begin by reviewing the concept of signatures and EMD in the next section. Then we will outline an EMD-based LDB algorithm in Sect. 12.3. Next, in Sect. 12.4, we will compare performance of all three LDB algorithms on synthetic datasets using various base classifiers. Finally, we conclude in Sect. 12.5 with a summary of performance.

12.2 Earth Mover's Distance

A *signature* represents a set of clusters of feature vectors, say, in $\mathbb{R}^d$. Each such cluster is represented by its mean, $\mathbf{m}_j$, of vectors belonging to that cluster and the weight (or importance) of that cluster, $w_{\mathbf{m}_j}$. The number of clusters in a signature varies with the complexity of the object being represented. Signatures are generalized histograms. A histogram is a fixed partitioning of the underlying space with cluster centers defined as the central value in each bin. The weight of each cluster is the percentage of points in the bin. The flexibility provided by signatures is the ability to place the "bins" where the data is located. For example, representation of data that exists on a curved manifold might require a relatively fine partitioning of the space to achieve a histogram that captures the distribution of the data. However, a signature representation is likely to be much more efficient since we are not required to partition the entire space, and feature clusters can be placed at ideal locations along the manifold. A comparison of histograms and signatures is detailed in [8]. Unfortunately, most dissimilarity measures cannot be applied to signatures. This is because they rely on direct correspondence between bins. That is, they can be used for histograms that contain the same number of bins. This is, however, not guaranteed with signatures. EMD, on the other hand, is designed for use with signatures.

EMD was first introduced by Rubner, et al. [8] for retrieval of color and textured images. It has several properties that have many advantages over other distance measures:

- Applies to signatures
- Naturally reflects nearness
- Allows for partial matching
- Is a metric (if total weights of two signatures are equal and cost is a metric)

EMD has the intuitive interpretation of the minimum amount of work required to move piles of soil (or earth) into holes. The location and size of the piles of soil are

represented by cluster centers and weights of a signature, respectively. Similarly, the other signature represents the location and size of the holes to be filled. Formally, if we let

$$P = \{(\mathbf{p}_1, w_{\mathbf{p}_1}), \ldots, (\mathbf{p}_m, w_{\mathbf{p}_m})\},$$
$$Q = \{(\mathbf{q}_1, w_{\mathbf{q}_1}), \ldots, (\mathbf{q}_n, w_{\mathbf{q}_n})\}$$

be our two signatures and $C = [c_{ij}]$ the cost matrix where c_{ij} represents the cost of moving one unit of mass from the ith cluster in P to the jth cluster in Q, then the EMD algorithm seeks the flow $F = [f_{ij}]$ that minimizes the work

$$W(P,Q,F) := \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij},$$

subject to the constraints:

$$f_{ij} \geq 0 \qquad 1 \leq i \leq m, 1 \leq j \leq n;$$
$$\sum_{j=1}^{n} f_{ij} \leq w_{\mathbf{p}_i} \qquad 1 \leq i \leq m;$$
$$\sum_{i=1}^{m} f_{ij} \leq w_{\mathbf{q}_j} \qquad 1 \leq j \leq n;$$
$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = \min \left(\sum_{i=1}^{m} w_{\mathbf{p}_i}, \sum_{j=1}^{n} w_{\mathbf{q}_j} \right).$$

Once the optimal flow F is found, EMD is the resulting work normalized by the total flow:

$$\mathrm{EMD}(P,Q) := \frac{W(P,Q,F)}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}}.$$

This normalization is necessary to avoid favoring smaller signatures if the two signatures have different total weights.

The optimal flow is found by solving the well-known transportation problem, or *the Monge-Kantorovich mass transportation problem* [5]. Typically, this requires the use of a linear programming such as the simplex method to solve for the optimal flow. A detailed explanation of the simplex method can be found in [5]. However, there are a few situations where fast algorithms that do not require linear programming can be used.

In particular, for the one dimensional case where the cost is the Euclidean distance and the signatures have equal total weights, w_Σ, Rubner and Tomasi [7, Sect. 2.3.1] showed that the EMD can be directly calculated by

$$\mathrm{EMD}(P,Q) = \frac{\sum_{k=1}^{m+n-1} |\hat{p}_k - \hat{q}_k|(r_{k+1} - r_k)}{w_\Sigma},$$

where $r_1, r_2, \ldots, r_{m+n}$ is the sorted list of

$$p_1, p_2, \ldots, p_m, q_1, q_2, \ldots, q_n,$$

and

$$\hat{p}_k := \sum_{p_i \le r_k} w_{p_i}, \qquad \hat{q}_k := \sum_{q_i \le r_k} w_{q_i}.$$

The algorithm relies on the fact that the minimum work between two one dimensionals distributions is known to be the L_1 distance between the cumulative distribution functions (cdfs), as discussed by [7, Sect. 2.3.1]. We note that the L_1 distance between two cdfs is a special instance of the so-called $\bar{\rho}$ or Ornstein distance [1]; see also [6] on the deep relationship between EMD and the Marrows distance often used in statistics. The following theorem is presented for convenience:

Theorem 1. *Define the empirical cdfs of 1D signatures P and Q as*

$$P(t) := \begin{cases} 0 & t \in (-\infty, p_{(1)}), \\ \sum_{i=1}^{k} w_{p_{(i)}} & t \in [p_{(k)}, p_{(k+1)}),\ 1 \le k \le m-1, \\ \sum_{i=1}^{m} w_{p_{(i)}} & t \in [p_{(m)}, \infty), \end{cases}$$

$$Q(t) := \begin{cases} 0 & t \in (-\infty, q_{(1)}), \\ \sum_{j=1}^{k} w_{q_{(j)}} & t \in [q_{(k)}, q_{(k+1)}),\ 1 \le k \le n-1, \\ \sum_{j=1}^{n} w_{q_{(j)}} & t \in [q_{(n)}, \infty), \end{cases}$$

where $\{p_{(i)}\}$ and $\{q_{(j)}\}$ are sorted versions (in nondecreasing order) of $\{p_i\}$ and $\{q_j\}$, respectively. If P and Q have equal total weights $\sum_{i=1}^{m} w_{p_i} = \sum_{j=1}^{n} w_{q_j} =: w_\Sigma$, then

$$\mathrm{EMD}(P,Q) = \frac{\int_{-\infty}^{\infty} |P(t) - Q(t)|\,\mathrm{d}t}{w_\Sigma}.$$

Throughout this chapter we will be using this fast 1D version of EMD, which restricts our cost function to L_1 distance. We could use a simplex solver, which would allow other cost functions, but the use of the simplex solver greatly impacts speed. The worst-case computational cost for a simplex solver is exponential, but the use of a transportation-simplex solver and a good initial solution (close to an optimal solution) greatly improves performance. Rubner and Tomasi report [7, Sect. 2.3] an empirical performance of $O(n^3 \log n)$ in the case when both P and Q have $n = m$ clusters. This is opposed to the fast 1D EMD solver which has a

computational cost of $O(n \log n)$. Clustering helps lighten the computational burden of the transportation-simplex solver, but introduces complexity to the algorithm (in the form of clustering parameters) and presents variability of results. With 1D EMD we can completely avoid clustering the data.

12.3 EMD-Based Local Discriminant Basis

Generally, the LDB algorithm is broken into the following steps:

Algorithm 2. *Given a training dataset $\mathscr{T}$ that consists of C classes of signals $\{\{\mathbf{x}_i^{(c)}\}_{i=1}^{N_c}\}_{c=1}^{C}$:*

- *Choose a dictionary and specify the maximum level of decomposition.*
- *Expand all the signals into tree-structured subspaces.*
- *For each class, compute a time–frequency map.*
- *Use the time–frequency maps to compute the discriminating power of each coordinate.*
- *Prune the tree by examining the discriminating power of each subspace.*
- *Order the basis vectors by their discriminating power.*

This process is made efficient by the speed with which the signals can be expanded into the selected dictionary, and by exploiting the tree structure in the pruning process and using additive discriminant measures for comparison.

Definition 1. Let $\mathbf{p}$ and $\mathbf{q}$ be any two vectors in $\mathbb{R}^n$. A *discriminant measure*, $D(\mathbf{p},\mathbf{q})$, is a map $D : (\mathbb{R}^n \times \mathbb{R}^n) \to \mathbb{R}$ such that

- $D(\mathbf{p},\mathbf{p}) = 0$.
- (Nonnegative) $D(\mathbf{p},\mathbf{q}) \geq 0$ for all $\mathbf{p}$, $\mathbf{q}$ in $\mathbb{R}^n$.

A discriminant measure is said to be *additive* if

$$D(\mathbf{p},\mathbf{q}) = \sum_{i=1}^{n} D(p_i, q_i).$$

For our EMD-based LDB, we construct our time–frequency map by collecting signatures for each coordinate of each class in a subspace. Specifically, for coordinate l of class c in subspace $\Omega_{j,k}$, our signatures are

$$s_{(j,k,l)}^{(c)} = \left\{ \left(\mathbf{w}_{j,k,l} \cdot \mathbf{x}_i^{(c)}, 1/N_c \right) \right\}_{i=1}^{N_c}. \tag{12.1}$$

The collection of these signatures form our time–frequency map. In this formulation we have chosen our samples for each class to be equally weighted, which is reasonable since we are not assuming or attempting to compute the relative importance of a particular signal to its class or its overall ability to discriminate. To efficiently evaluate the discriminant power of a coordinate or subspace, we define our additive measure using EMD as follows.

Definition 2. Let $s^{(c)}_{(j,k,l)}$ be the signature for training signals belonging to class c at level j, block k, and position l. Then the *EMD distance between classes* is defined as the sum of the pairwise EMD distances:

$$\mathscr{D}(j,k,l) := \sum_{m=1}^{C-1} \sum_{n=m+1}^{C} \text{EMD}\left(s^{(m)}_{(j,k,l)}, s^{(n)}_{(j,k,l)}\right). \tag{12.2}$$

We refer to $\mathscr{D}(j,k,l)$ as the discriminant power of the coordinate. The *discriminant power of a subspace* is then

$$\Gamma(j,k) := \sum_{l=0}^{2^{n_0-j}-1} \mathscr{D}(j,k,l), \quad 0 \le j \le J \le n_0; 0 \le k \le 2^j - 1, \tag{12.3}$$

where the length of each signal in the dataset is assumed to be $n = 2^{n_0}$, and J is the maximum depth of decomposition set by the user.

We also use the notation $\mathscr{D}(\Omega_{j,k}) := \Gamma(j,k)$ to emphasize that $\Gamma(j,k)$ is a discriminant measure for the subspace $\Omega_{j,k}$. Often the sum in (12.3) is truncated; more precisely, we only sum the k_0 largest values from each subspace rather than summing all the 2^{n_0-j} values as (12.3). (Note that for a certain j for k_0 set by the user, we could have $k_0 > 2^{n_0-j}$. In that case, we sum all the 2^{n_0-j} values.) This helps with situations where there are many weak coordinates summing to a large value.

Note that $\mathscr{D}(\Omega_{j,k})$ is an additive discriminant for subspace $\Omega_{j,k}$ since it has been defined as the sum of the discriminant powers of the subspace coordinates. The benefit of an additive discriminant measure comes in the pruning process. Using the tree-structure notation shown in Fig. 12.1, pruning starts at the base of the tree and is conducted by the following rule:

Algorithm 3. *Let* $0 \le j \le J \le n_0$ *and* $0 \le k \le 2^j - 1$. *If* $\mathscr{D}(\Omega_{j,k}) \ge \mathscr{D}(\Omega_{j+1,2k} \cup \Omega_{j+1,2k+1})$, *then select* $\Omega_{j,k}$ *over* $\Omega_{j+1,2k} \cup \Omega_{j+1,2k+1}$; *otherwise, select* $\Omega_{j+1,2k} \cup \Omega_{j+1,2k+1}$.

If the measure $\mathscr{D}(\Omega_{j+1,2k} \cup \Omega_{j+1,2k+1})$ is additive, it can be efficiently computed as

$$\mathscr{D}(\Omega_{j+1,2k}) + \mathscr{D}(\Omega_{j+1,2k+1}),$$

which only requires the addition of the discriminant measure for the individual subspaces to obtain the measure of their union.

Substituting our signature time–frequency map, (12.1), and EMD discriminant measure, (12.3), into Algorithm 2, we obtain our EMD-based LDB algorithm. This algorithm benefits from the adaptive structure of signatures and the robustness of EMD while remaining computationally fast and capable of detecting fine differences with few parameters. Further, our signatures can be quickly updated to incorporate new information. This becomes important when we consider situations where training data is limited and/or the data is noisy. Here incremental learning is important so that new information can be incorporated without complete retraining

and better training samples can be used as they are available. We can avoid the need to retrain on the entire set by initializing our algorithm with a small training set and storing the signature map constructed above with the modification that we store the sum of the sample values and the number of samples in each cluster as our features and weights. This allows us to update the mean features and weights without storing any extra information. The signature's mean value and weight are calculated during the computation of the EMD. Any new data that we wish to incorporate can easily be added by decomposing it into the selected dictionary and incorporating it into our stored signature map. To accommodate limited memory resources and reduce time to extend signature storage, a specified limit can be set for the signature length during initialization, so that memory can be preallocated. If a signature has reached the storage capacity specified or is within a threshold distance to a cluster in the signature, then we can use our grouping technique which is similar to clustering.

Algorithm 4. *Let m^* be our capacity and $m < m^*$. Given signature*

$$S = \{(s_1, w_{s_1}), \ldots, (s_m, w_{s_m})\},$$

where s_i is the sum of each cluster value and w_{s_i} is the number of samples in each cluster for $i \in \{1, \ldots, m\}$. Let $0 \leq \tau$ be our grouping threshold and s_{new} a new sample:

- *If there exists $s_k, k \in \{1, \ldots, m\}$ such that $|s_{\text{new}} - s_k| < \tau$, then set $s_k = s_k + s_{\text{new}}$ and $w_{s_k} = w_{s_k} + 1$.*
- *Else add new sample to the end of the signature:*

$$S = \{(s_i, w_{s_1}), \ldots, (s_m, w_{s_m}), (s_{\text{new}}, 1)\}.$$

Although we are only working in 1D, Algorithm 4 works in any dimension as long as our signatures are not at capacity, $m = m^*$. When a signature reaches capacity we reduce the signature size by combining the closest clusters of the signature. Having our signature evolve in this manner means that our capacity acts as a resolution parameter and our grouping threshold as a sensitivity parameter. In a situation with a complex structure over a relatively large span, our capacity limit may force a less than ideal signature resolution reducing the signature's ability to describe the structure. Therefore, considerations for the training data must be taken when choosing signature capacity.

12.4 Local Discriminant Basis Algorithm Performance

The purpose of constructing a feature space is to pull out the important properties of the datasets for the purposes of discrimination or compression. A good feature space should provide dimension reduction and/or improved classifier performance. LDB algorithms naturally provide dimension reduction by expanding data into an

orthonormal basis followed by selecting the most discriminant basis coordinates. In practice, relatively few LDB vectors are usually needed for discrimination. To evaluate our algorithm on increasing classifier performance, we use two different base classifiers, linear discriminant analysis (LDA) and classification tree (CT); see, e.g., [4, Sects. 4.3 and 9.2] for the details of these classifiers. These two classifiers construct distinct decision boundaries, which gives us an indication of the complexity of the separation. LDA seeks an optimally separating hyperplane as its decision boundary. A classification tree seeks an optimal partitioning of each coordinate.

For comparison, we have three LDB algorithms available: time–frequency LDB algorithm (LDBK), epdf-based LDB algorithm (LDBKASH), and EMD-based LDB algorithm (LDBKEMD). Each algorithm is analyzing a different quantity to determine class separability. For LDBK we are concerned with normalized coefficient energy. For LDBKASH we measure separation of coefficients' epdfs. And for LDBKEMD we are computing the separation of the empirical cumulative distribution functions (ecdfs). All three are analyzing the same information, just in different ways. As we will see in the examples below, how you analyze the information greatly impacts the quality of the selected feature space. Our LDBKASH and LDBKEMD algorithms are using distribution information and not a single statistic for each coordinate. So, they are able to incorporate information about the statistical distribution of the coefficients into the selection process. This allows for the detection of subtler differences that can be lost in the case of a single statistic. However, estimating reliable epdfs using a noisy dataset with a limited number of samples is quite difficult.

To compare the performance of the different LDB algorithms, we conduct classification experiments on four different synthetic signal datasets. The first two datasets (triangular waveforms and shape waveforms) were used by Saito [9] to demonstrate the benefit of using the original LDB algorithms for classification. Our third dataset is a variation of the shape waveforms dataset constructed for subtle differences between classes. The last dataset looks at classes that differ in frequency content only. Each dataset contains three different classes of signals. For each class, we generated 100 signals to use for training each LDB algorithm. Another 1,000 signals are generated to test the constructed LDB feature spaces. For each LDB algorithm, we set the parameter $k_0 = 10$, i.e., we evaluate the goodness of each subspace using the top ten most discriminant coordinates. For each dataset, classifiers are trained using the top ten most discriminant features from each LDB algorithm. We repeat this process of generating training and test sets, computing features spaces, and applying classifiers ten times. The mean and standard deviation of the misclassification over the ten trials are presented in Table 12.1. As for the LDBK and LDBKASH algorithms, we use the symmetric relative entropy as the discriminant measure $D(\mathbf{p},\mathbf{q})$ throughout this article, i.e.,

$$D(\mathbf{p},\mathbf{q}) = J(\mathbf{p},\mathbf{q}) := \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} + q_i \log \frac{q_i}{p_i},$$

Table 12.1 Table of training and test waveforms misclassification for all examples discussed

	LDA		CT	
	Train	Test	Train	Test
Triangle waveform classification				
STD	11.40 % ± 1.10	22.18 % ± 1.18	3.93 % ± 0.78	30.58 % ± 1.82
LDBK	14.13 % ± 2.26	17.08 % ± 1.05	5.87 % ± 0.88	23.48 % ± 1.98
LDBKASH	13.17 % ± 2.18	16.60 % ± 0.85	4.90 % ± 1.31	22.36 % ± 1.34
LDBKEMD	13.90 % ± 1.75	16.38 % ± 0.84	6.27 % ± 1.91	23.82 % ± 2.10
Shape waveform classification				
STD	0.27 % ± 0.41	6.94 % ± 0.87	0.63 % ± 0.60	6.58 % ± 0.77
LDBK	4.10 % ± 2.62	4.62 % ± 1.66	0.87 % ± 0.59	3.96 % ± 1.27
LDBKASH	1.67 % ± 0.92	2.54 % ± 0.42	0.67 % ± 0.61	3.50 % ± 1.23
LDBKEMD	2.07 % ± 0.78	3.07 % ± 0.68	0.57 % ± 0.42	2.98 % ± 0.74
Bell waveform classification				
STD	5.50 % ± 1.57	28.93 % ± 4.81	2.67 % ± 1.31	20.78 % ± 1.80
LDBK	34.10 % ± 17.91	36.30 % ± 17.00	7.33 % ± 3.68	30.38 % ± 10.19
LDBKASH	26.07 % ± 6.35	28.40 % ± 5.87	5.00 % ± 1.61	24.38 % ± 4.57
LDBKEMD	16.33 % ± 6.25	19.40 % ± 5.81	4.27 % ± 0.81	17.91 % ± 1.66
Chirp waveform classification				
STD	0.00 % ± 0.00	0.00 % ± 0.00	1.63 % ± 0.81	18.91 % ± 1.15
LDBK	4.33 % ± 4.26	6.92 % ± 6.07	4.07 % ± 2.57	20.36 % ± 9.93
LDBKASH	18.37 % ± 9.39	22.86 % ± 9.26	5.93 % ± 2.02	32.42 % ± 7.74
LDBKEMD	0.00 % ± 0.00	0.00 % ± 0.00	0.00 % ± 0.00	1.21 % ± 0.51

Each example has results for all LDB algorithm using linear discriminate analysis and classification tree classifiers. STD above indicates the use of the standard coordinate system for representing signals, i.e., the raw signals are directly fed to the classifiers

Example 1. Triangular Waveforms

This example was originally examined in [2]. Later, Saito [9] extended the length of the signals from 21 to 32, so that the signals are of dyadic length and could be used to evaluate the performance of LDB. The example consists of three classes of signals which are formed from a convex linear combination of triangular waveforms. Specifically, the classes of signals are generated by the following formulas:

$$x^{(1)}(i) = uh_1(i) + (1-u)h_2(i) + \varepsilon(i) \qquad \text{for class 1,} \tag{12.4}$$

$$x^{(2)}(i) = uh_1(i) + (1-u)h_3(i) + \varepsilon(i) \qquad \text{for class 2,} \tag{12.5}$$

$$x^{(3)}(i) = uh_2(i) + (1-u)h_3(i) + \varepsilon(i) \qquad \text{for class 3,} \tag{12.6}$$

where $i = 1, \ldots, 32$, $h_1(i) = \max(6 - |i-7|, 0)$, $h_2(i) = h_1(i-8)$, $h_3(i) = h_1(i-4)$, u is a uniform random variable on $(0,1)$, and $\varepsilon(i)$'s are the i.i.d. standard normal variates. Five sample waveforms from each class are shown in Fig. 12.2. This example is convenient for performance evaluation because Breiman et al. [2]

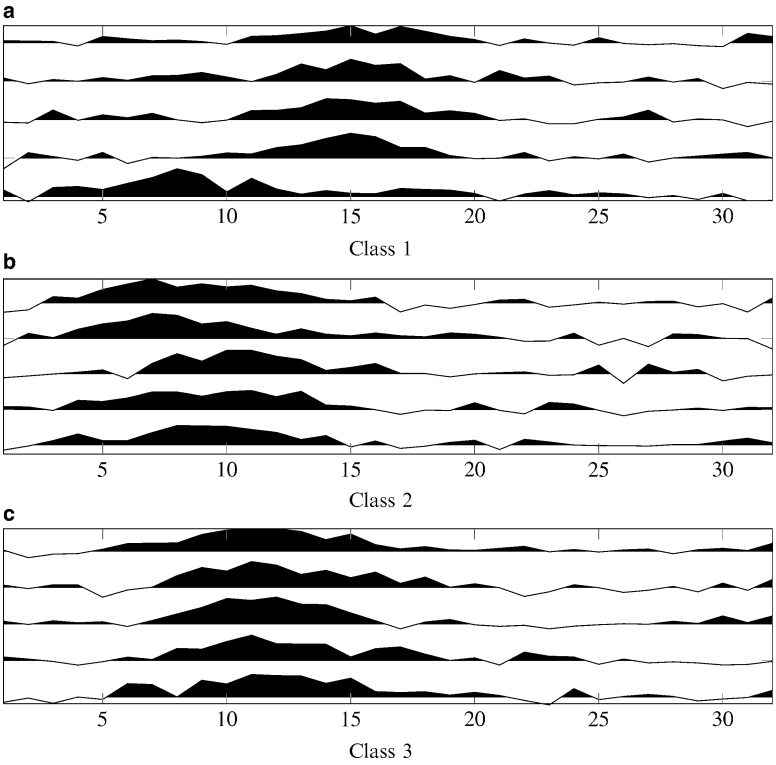

Fig. 12.2 Five sample waveforms from each class of triangular waveform dataset. Waveforms are generated using (12.4)–(12.6)

computed the Bayes error rate to be 14 %, which gives us an ideal classifier performance to expect. We use the wavelet packet dictionary with a 6-tap coiflet filter [3, Sect. 8.2] to compute the LDB.

All three algorithms exhibit similar classification performance and had similar basis selections. The LDA classifier performs the best with both training and test classification results near the Bayes error rate, see Table 12.1. If we look at the most discriminating basis vectors selected, Fig. 12.3a, we can identify the distinction that is being used to discriminate between the classes. The vectors with the greatest discriminant power are concentrated near the peaks of h_1, h_2, and h_3. The basis vectors are detecting the presence of the distinct characteristics of each class, i.e., the triangular peaks of h_1, h_2, and h_3. The coefficient plot for all three algorithms are similar, so we only show one in Fig. 12.3b. We see that each class lies on a linear manifold segment. The difficult signals to classify lie at the intersections of these linear manifold segments. These correspond to signals where one triangular peak is much more prominent. For this example, there is no extra benefit to examining the statistical distribution of the coefficients for each class. The time–frequency energy maps provide, as computed by LDBK, sufficient information for discrimination.

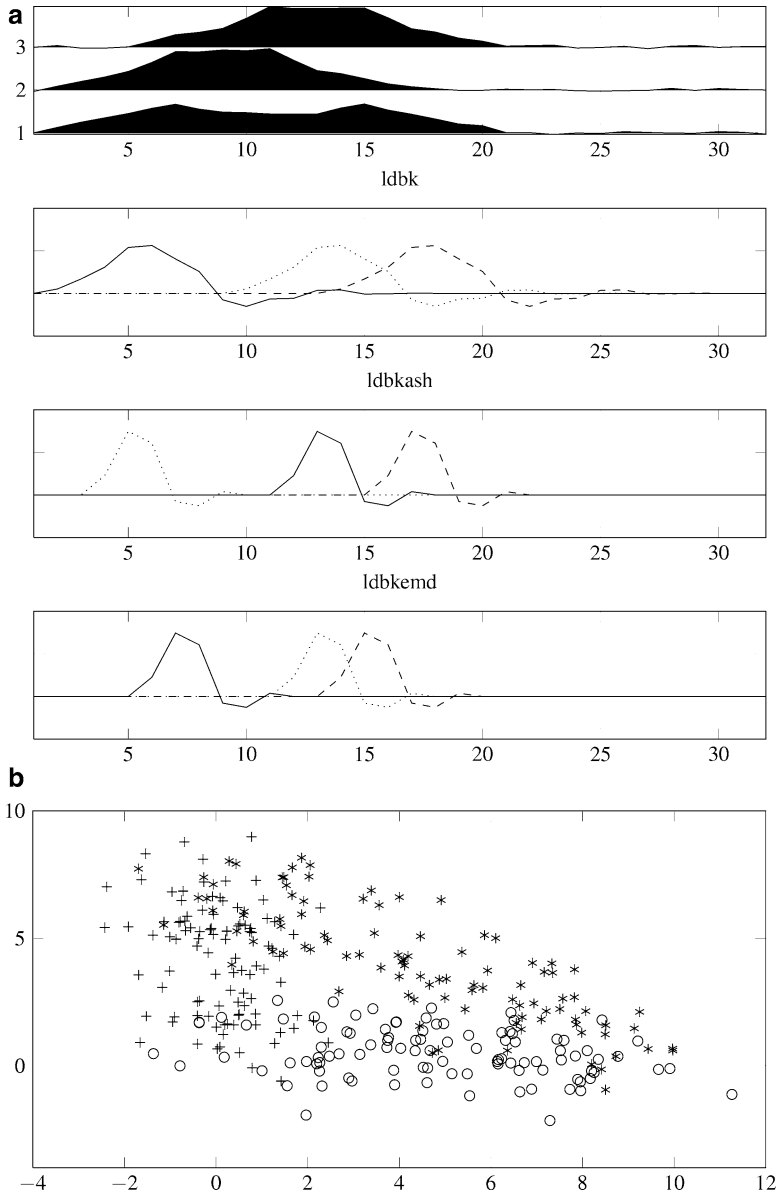

Fig. 12.3 (**a**) In the upper plot the mean training waveform for each of the triangular waveform dataset is shown. In the plot below it, the three most discriminant LDB vectors are shown for each LDB algorithm. (**b**) A scatter plot of the coefficients of the training signals in the top two most discriminating LDBK coordinates. The symbols, "*," "o," "+," represent class 1, class 2, class 3 signals, respectively

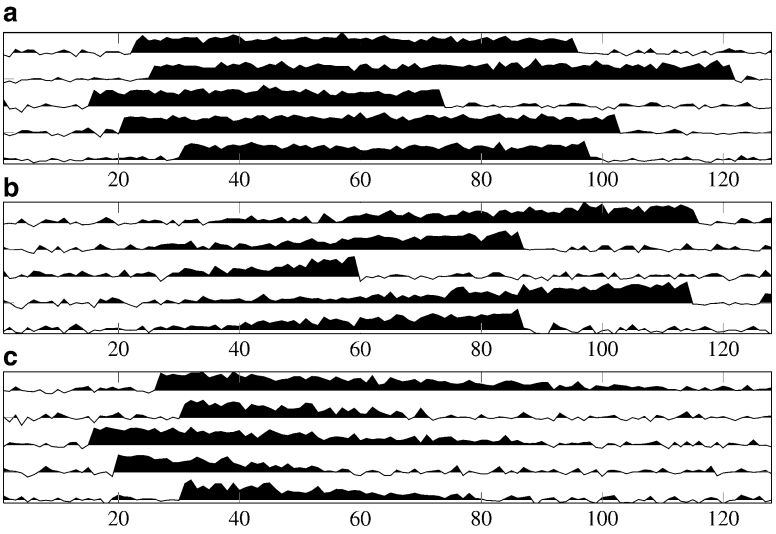

Fig. 12.4 Five sample waveforms from each class of the shape waveform dataset

Example 2. Shape Waveforms

Our second dataset is comprised of signal classes of different shapes (cylinder, bell, and funnel). All classes of signals are of finite duration, varied over when they appear. The cylinder signal class is a flat-amplitude (i.e., boxcar) signal. The bell signal class has a linearly increasing amplitude. And the funnel signal class has a linearly decreasing amplitude. More precisely, our signal classes are generated by the following formulae:

$$
\begin{aligned}
c(i) &= (6+\eta)\cdot\chi_{[a,b]}(i)+\varepsilon(i) && \text{cylinder,}\\
b(i) &= (6+\eta)\cdot\chi_{[a,b]}(i)\cdot(i-a)/(b-a)+\varepsilon(i) && \text{bell,}\\
f(i) &= (6+\eta)\cdot\chi_{[a,b]}(i)\cdot(b-i)/(b-a)+\varepsilon(i) && \text{funnel,}
\end{aligned}
$$

where $i = 1,\ldots,128$, a is an integer-valued uniform random variable on the interval [16,32], $b-a$ also obeys an integer-valued uniform distribution on [32,96], η and $\varepsilon(i)$'s are the i.i.d. standard normal variates, and $\chi_{[a,b]}(i)$ is the characteristic, or indicator, function on the interval $[a,b]$. Five example waveforms from each class can be seen in Fig. 12.4. We use the wavelet packet coefficients from an 18-tap coiflet filter [3, Sect. 8.2] to compute the LDB.

The ten most discriminating LDB vectors for each algorithm are shown in Fig. 12.5. Classification performance, shown in Table 12.1, for each algorithm is similar with a slight benefit to using LDBKASH or LDBKEMD. However, the top LDB vectors selected do vary for each algorithm. For all the algorithms, within the ten most discriminating vectors selected, there are features concentrated at

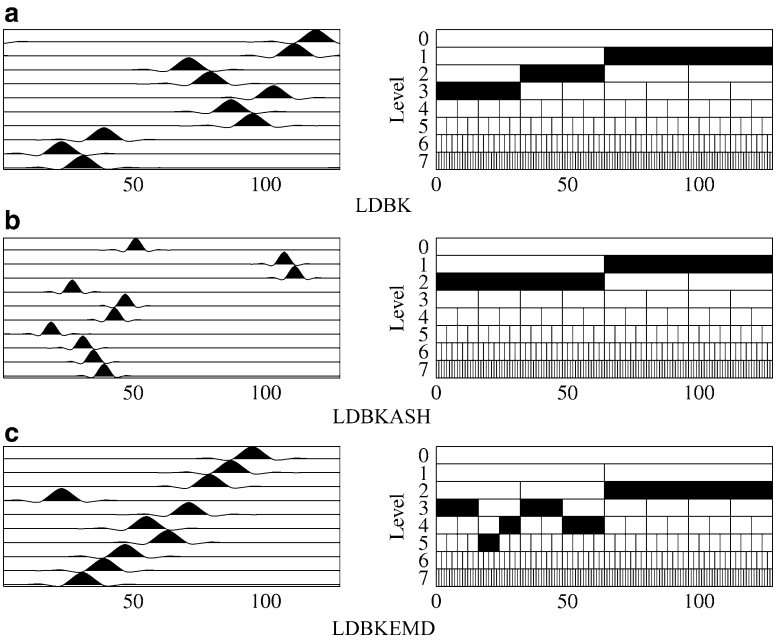

Fig. 12.5 Top ten LDB vectors selected for the shape waveform using different LDB algorithms. The subfigures in the right column show the selected subspaces indicated in *black*

the beginning and end of where we expect the cylinder, bell, or funnel to be. For LDBKEMD, we note that unlike LDBK and LDBKASH, there are vectors concentrated in the middle. While the ends emphasize the regions where we are likely to see the most dramatic change in signal characteristics, our signals do vary within the interval $[a,b]$. For our next example we will try to emphasize this by making our classes more similar.

Example 3. Bell Waveforms

In the previous example we noted that the different LDB algorithms selected somewhat different basis vectors. However, the differences had little impact on the classification performance since all the LDB algorithms selected features that focused on regions where there was a great deal of change between classes. For this example, we attempt to make the shape distinction more subtle by considering three bells with varying slopes. Specifically, the signals are generated by the following formulae:

$$b_1(i) = \chi_{[a,b]}(i) \cdot (i-a) \cdot S_1 + \varepsilon(i) \qquad \text{bell 1,}$$

$$b_2(i) = \chi_{[a,b]}(i) \cdot (i-a) \cdot S_2 + \varepsilon(i) \qquad \text{bell 2,}$$

$$b_3(i) = \chi_{[a,b]}(i) \cdot (i-a) \cdot S_3 + \varepsilon(i) \qquad \text{bell 3,}$$

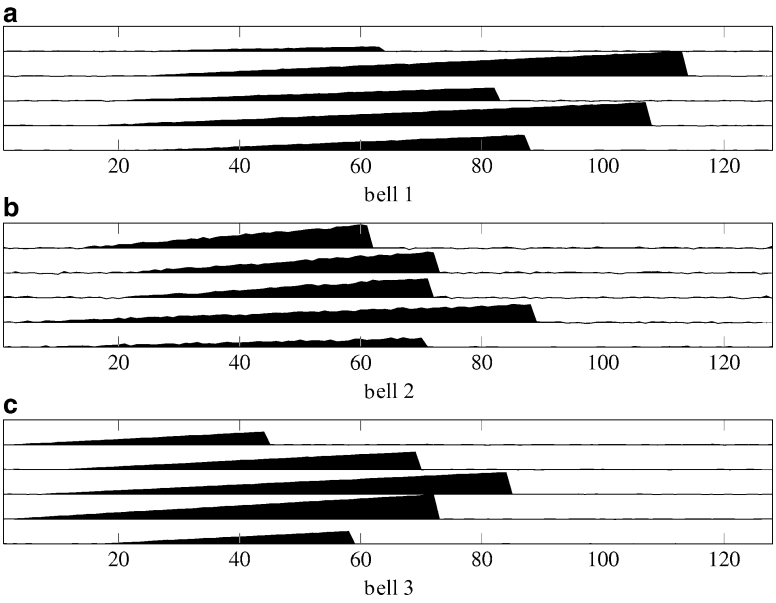

Fig. 12.6 Five sample waveforms from each class of the Bell waveform dataset

where $i = 1,\ldots,128$, a is an integer-valued uniform random variable on the interval [16,32], $b-a$ also obeys an integer-valued uniform distribution on [32,96], $\varepsilon(i)$'s are the i.i.d. standard normal variates, and $\chi_{[a,b]}(i)$ is the characteristic, or indicator, function on the interval $[a,b]$. The slopes (S_1, S_2, and S_3) are normal random variables with standard deviation 0.2 and respective means of 1, 0.5, and 2. Five sample waveforms from each class can be seen in Fig. 12.6. Visually it is quite difficult to distinguish between the classes of signals. For training, the LDB is computed from the wavelet packet coefficients with a 18-tap coiflet filter [3, Sect. 8.2].

Looking at the classification performance, shown in Table 12.1, we see a clear performance increase between using energy versus the distribution of energy. The LDBKASH and LDBKEMD algorithms show significantly lower average misclassification rates and standard deviation than LDBK. Looking at the ten selected basis vectors shown in Fig. 12.7, we see that the LDBKEMD vectors are varied in position across the length of the signal with the center location being the most important. The LDBKASH vectors have a narrower support and are concentrated toward the center of the signal. The LDBK vectors are concentrated toward the end of the signal where we would expect the greatest impact from the varying slope. However, there is also a chance of the signal not being there due to the randomness in the parameter b, the signal ending position. From the coefficient plot, Fig. 12.8, we can see that the top 3 features for LDBKASH and LDBKEMD correspond to the change in slope for each class. Bell 2 had the smallest slope, and we see in the coefficient plot, indicated with the "o" marker, that it is at the bottom.

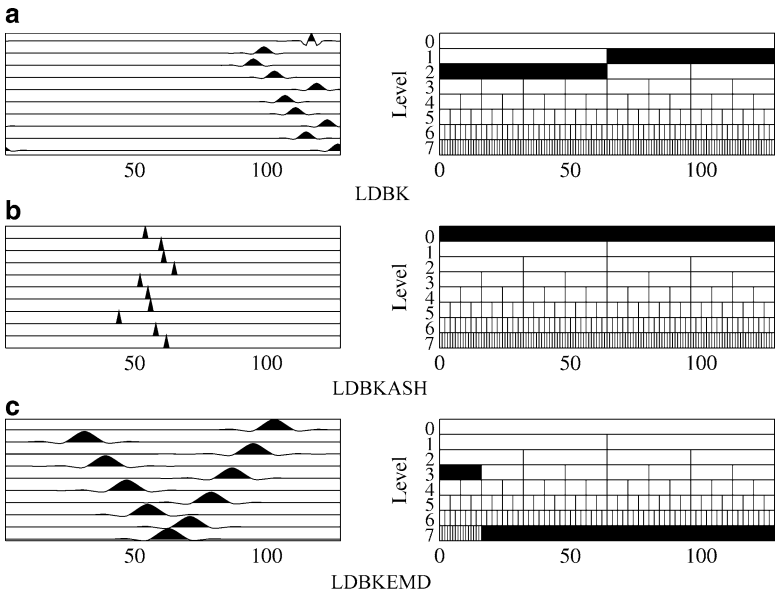

Fig. 12.7 Top ten LDB vectors selected for the bell waveforms using different LDB algorithms. The subfigures in the right column show the selected subspaces indicated in *black*

Then we see that bell 1, indicated with the "*" marker, follows after bell 2 with the intermediate slope. And last we see bell 3, indicated with the "+" marker, which had the largest slope.

Example 4. Chirp Waveforms
For our final example we will look at the ability of our LDB algorithms to distinguish between frequency differences in signals. Our dataset will consist of three different quadratic chirp waveforms with additive Gaussian noise. All chirps start with 200-Hz oscillations, sweep down to varied minimum frequency ranges, and go up to the 200-Hz level again in the end as the spectrogram in Fig. 12.9 shows. More specifically,

$$c_1(i) = \cos\left(2\pi\left[\frac{80}{3}t(i)^3 + 120t(i)\right]\right) + \varepsilon(i) \qquad \text{chirp 1,}$$

$$c_2(i) = \cos\left(2\pi\left[\frac{50}{3}t(i)^3 + 150t(i)\right]\right) + \varepsilon(i) \qquad \text{chirp 2,}$$

$$c_3(i) = \cos\left(2\pi\left[\frac{20}{3}t(i)^3 + 180t(i)\right]\right) + \varepsilon(i) \qquad \text{chirp 3,}$$

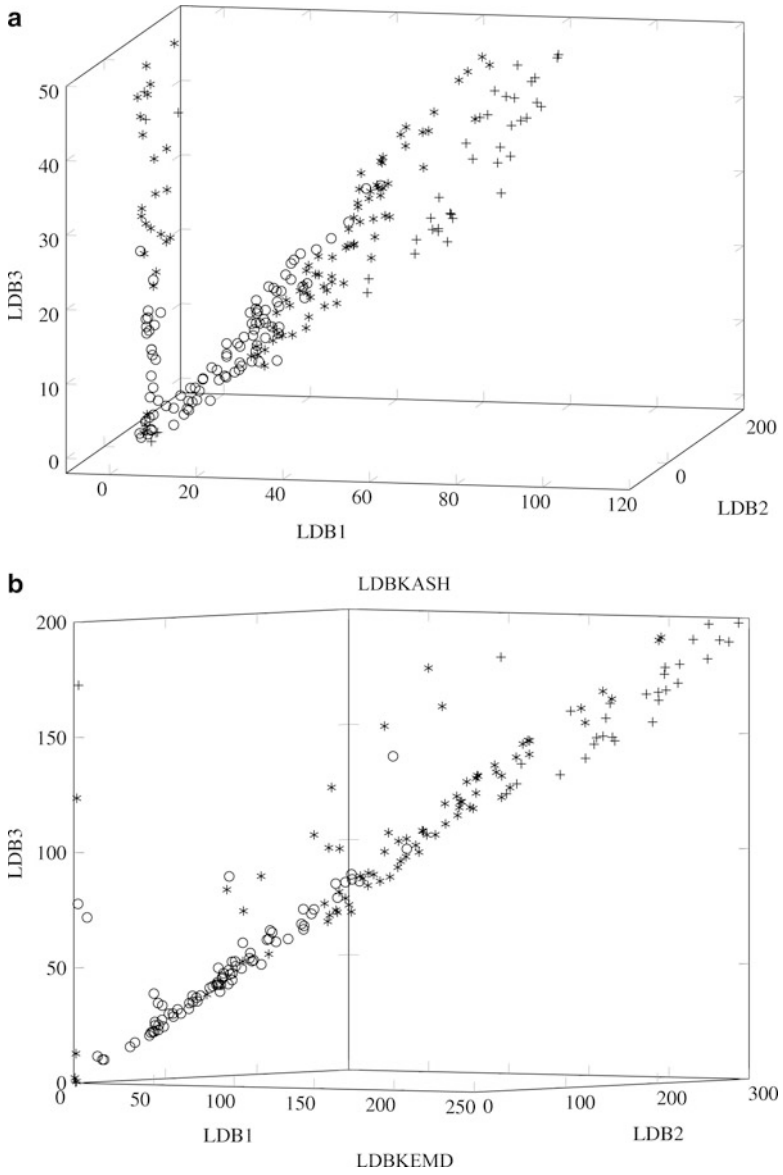

Fig. 12.8 The distribution of the coefficients in the top three LDB coordinates for bell waveform training and test datasets using (**a**) LDBKASH and (**b**) LDBKEMD algorithms. The symbols, "*," "o," "+," represent bell 1, bell 2, bell 3 signals, respectively

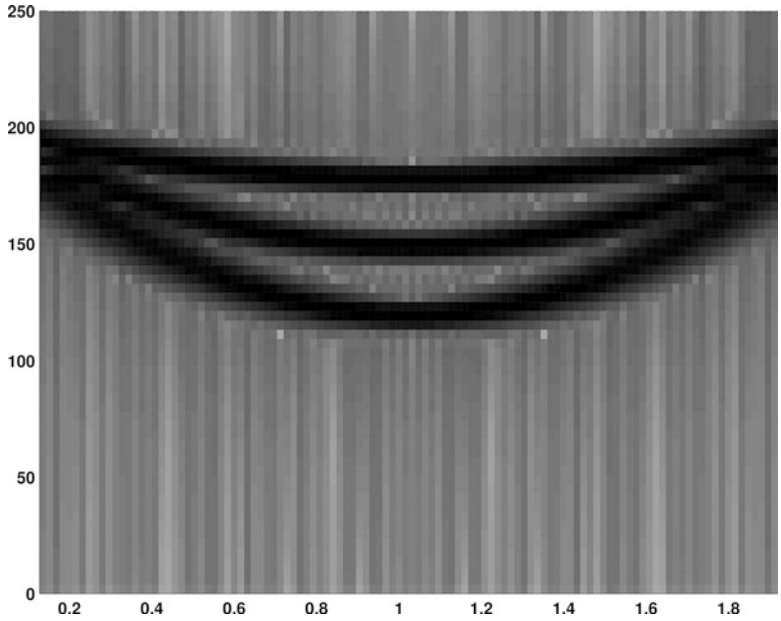

Fig. 12.9 Spectrogram of chirp waveforms for chirp waveform dataset. Horizontal axis is time (seconds). Vertical axis is frequency (Hz)

where $i = 1, \ldots, 1024$, $t(i) = i/512 - 1$, and $\varepsilon(i)$'s are i.i.d. standard normal variates. For training, the LDB is computed from the wavelet packet coefficients with a 18-tap coiflet filter.

From the result shown in Table 12.1, we see that LDBKEMD performs very well with zero misclassification for an LDA classifier. The LDBKASH algorithm performs the worst with a test misclassification rate around 23 % with an LDA classifier. For LDBK the test misclassification rate is around 7 % with an LDA classifier. If we look at the resulting LDB selected for each algorithm and top ten LDB vectors, Fig. 12.10, we see that the LDBKASH algorithm selected the root level with the most discriminating features being concentrated toward the center of the signal. The LDBK algorithm also selected features concentrated toward the center, but with wider support. On the other hand, the features selected by the LDBKEMD algorithm are quite different. Some of the LDBKEMD vectors are concentrated around the beginning and ending locations while the others have much wider supports in time.

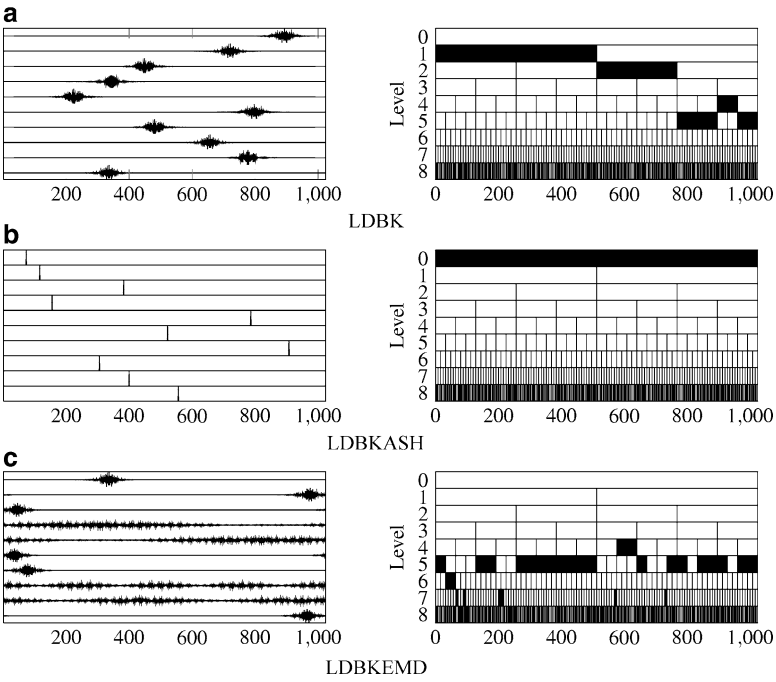

Fig. 12.10 Top ten LDB vectors selected for the chirp waveforms using different LDB algorithms. The subfigures in the right column show the selected subspaces indicated in *black*

12.5 Conclusion

LDB is an effective and computationally fast method for extracting discriminant features from signals. Furthermore, as we demonstrated in Sect. 12.4, the resulting feature space is interpretable; we know what is being used in each feature through the corresponding LDB basis vector and the expansion coefficients of the signals relative to that vector and therefore have a better understanding of when it will not be as effective and why. Also, we presented yet another LDB algorithm using a new discriminant measure based on signatures and EMD and demonstrated its capability of detecting features that can be missed using other versions of LDBs. Our EMD-based LDB can also be adapted to use new training data as it is provided. In comparison to LDBKASH, our LDBKEMD algorithm has fewer parameters to tweak and avoids the difficult task of estimating reliable epdfs, which make this new algorithm more robust. As demonstrated by our last two examples, it provides for better separation of classes with less training and a measure of discrimination that is less susceptible to outliers. However, as shown with our first two examples, in some situations expending extra effort, i.e., incorporating statistical behavior of the expansion coefficients, does not necessarily improve the performance.

Acknowledgments This research was partially supported by grants NSF DMS-0410406, ONR N00014-06-1-0615, N00014-07-1-0166, N00014-09-1-0041, and N00014-09-1-0318 as well as NSF VIGRE grant DMS-0135345. Further, support was provided from the Inhouse Laboratory Independent Research (ILIR) program at the Naval Surface Warfare Center, Panama City.

References

1. Basseville M (1989) Distance measures for signal processing and pattern recognition. Signal Process 18:349–369
2. Breiman L, Friedman JH, Olshen RA, Stone C (1993) Classification and regression trees. Chapman & Hall, Inc., New York
3. Daubechies I (1992) Ten lectures on wavelets. SIAM, Philadelphia, PA,
4. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer, Berlin
5. Hillier FS, Lieberman GJ (1995) Introduction to mathematical programming. McGraw-Hill, San Francisco
6. Levina E, Bickel P (2001) The earth mover's distance is the mallow distance: Some insights from statistics. In: Proceedings of 8th IEEE International Conference on computer vision, pp 251–256
7. Rubner Y, Tomasi C (2000) Perceptual metrics for image database navigation. Kluwer Academic Publishers, Boston
8. Rubner Y, Tomasi C, Guibas LJ (2000) The earth mover's distance as a metric for image retrival. Int J Comput Vis 40:99–121
9. Saito N (1994) Local feature extraction and its applications using a library of bases. Ph.D thesis. Department of Mathematics, Yale University
10. Saito N, Coifman RR (1995) Local discriminant bases and their applications. J Math Imag Vis 5:337–358
11. Saito N, Coifman RR, Geshwind FB, Warner F (2002) Discriminant feature extraction using empirical probability density estimation and a local basis library. Pattern Recogn 35:2841–2852
12. Scott DW (1985) Averaged shifted histograms: effective nonparametric density estimators in several dimensions. Ann Stat 13:1024–1040