Spectral clustering for speech separation

Francis R. Bach FRANCIS.BACH@MINES.ORG
INRIA - WILLOW Project-Team
Laboratoire d'Informatique de l'Ecole Normale Supérieure (CNRS/ENS/INRIA UMR 8548)
45, rue d'Ulm, 75230 Paris, France

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720, USA

Abstract

Spectral clustering refers to a class of recent techniques which rely on the eigenstructure of a similarity matrix to partition points into disjoint clusters, with points in the same cluster having high similarity and points in different clusters having low similarity. In this chapter, we introduce the main concepts and algorithms together with recent advances in learning the similarity matrix from data. The techniques are illustrated on the blind one-microphone speech separation problem, by casting the problem as one of segmentation of the spectrogram.

1. Introduction

Clustering has many applications in machine learning, exploratory data analysis, computer vision and speech processing. Many algorithms have been proposed for this task of grouping data into several subsets that share some common structure (see, e.g., Hastie et al. (2001), Mitchell (1997)). Two distinct approaches are very popular: (1) Linkage algorithms are based on thresholding pairwise distances and are best suited for complex elongated structures, but are very sensitive to noise in the data; (2) K-means algorithms, on the other hand, are very robust to noise but are best suited for rounded linearly separable clusters. Spectral clustering is aimed at bridging the gap between these approaches, providing a methodology for finding elongated clusters while being more robust to noise than linkage algorithms.

Spectral clustering relies on the eigenstructure of a similarity matrix to partition points into disjoint clusters, with points in the same cluster having high similarity and points in different clusters having low similarity. As presented in Section 2, spectral clustering can be cast as a relaxation of a hard combinatorial problem based on *normalized cuts*.

Most clustering techniques explicitly or implicitly assume a metric or a similarity structure over the space of configurations, which is then used by clustering algorithms. The success of such algorithms depends heavily on the choice of the metric, but this choice is generally not treated as part of the learning problem. Thus, time-consuming manual feature selection and weighting is often a necessary precursor to the use of spectral methods.

Several recent papers have considered ways to alleviate this burden by incorporating prior knowledge into the metric, either in the setting of K-means clustering (Wagstaff et al., 2001, Xing et al., 2003, Bar-Hillel et al., 2003) or spectral clustering (Yu and Shi, 2002,

Kamvar et al., 2003). In this chapter, we consider a complementary approach, providing a general framework for learning the similarity matrix for spectral clustering from examples. We assume that we are given sample data with known partitions and are asked to build similarity matrices that will lead to these partitions when spectral clustering is performed. This problem is motivated by the availability of such datasets for at least two domains of application: in vision and image segmentation, databases of hand-labeled segmented images are now available (Martin et al., 2001), while for the blind separation of speech signals via partitioning of the time-frequency plane (Brown and Cooke, 1994), training examples can be created by mixing previously captured signals.

Another important motivation for our work is the need to develop spectral clustering methods that are robust to irrelevant features. Indeed, as we show in Section 4.5, the performance of current spectral methods can degrade dramatically in the presence of such irrelevant features. By using our learning algorithm to learn a diagonally-scaled Gaussian kernel for generating the similarity matrix, we obtain an algorithm that is significantly more robust.

Our work is based on a cost function that characterizes how close the eigenstructure of a similarity matrix $\mathbf{W}$ is to a partition $\mathbf{E}$. We derive this cost function in Section 2. As we show in Section 2.5, minimizing this cost function with respect to the partition $\mathbf{E}$ leads to a new clustering algorithm that takes the form of weighted K-means algorithms. Minimizing them with respect to $\mathbf{W}$ yields a theoretical framework for learning the similarity matrix, as we show in Section 3. Section 3.3 provides foundational material on the approximation of the eigensubspace of a symmetric matrix that is needed for Section 4, which presents learning algorithms for spectral clustering.

We highlight one other aspect of the problem here—the major computational challenge involved in applying spectral methods to domains such as vision or speech separation. Indeed, in image segmentation, the number of pixels in an image is usually greater than hundreds of thousands, leading to similarity matrices of potential huge sizes, while, for speech separation, four seconds of speech sampled at 5.5 kHz yields 22,000 samples and thus a naive implementation would need to manipulate similarity matrices of dimension at least $22,000 \times 22,000$. Thus a major part of our effort to apply spectral clustering techniques to speech separation has involved the design of numerical approximation schemes that exploit the different time scales present in speech signals. In Section 4.4, we present numerical techniques that are appropriate for generic clustering problems, while in Section 6.3, we show how these techniques specialize to speech. The results presented in this chapter are taken from Bach and Jordan (2006).

2. Spectral clustering and normalized cuts

In this section, we present our spectral clustering framework. Following Shi and Malik (2000) and Gu et al. (2001), we derive the spectral relaxation through normalized cuts. Alternative frameworks, based on Markov random walks (Meila and Shi, 2002), on different definitions of the normalized cut (Meila and Xu, 2003), or on constrained optimization (Higham and Kibble, 2004), lead to similar spectral relaxations.

2.1 Similarity matrices

Spectral clustering refers to a class of techniques for clustering that are based on pairwise similarity relations among data points. Given a dataset $\mathcal{I}$ of P points in a space $\mathcal{X}$, we assume that we are given a $P \times P$ "similarity matrix" $\mathbf{W}$ that measures the similarity between each pair of points: $\mathbf{W}_{pp'}$ is large when points indexed by p and p' are preferably in the same cluster, and is small otherwise. The goal of clustering is to organize the dataset into disjoint subsets with high intra-cluster similarity and low inter-cluster similarity.

Throughout this chapter we always assume that the elements of $\mathbf{W}$ are nonnegative ($\mathbf{W} \geqslant 0$) and that $\mathbf{W}$ is symmetric ($\mathbf{W} = \mathbf{W}^\top$). Moreover, we make the assumption that the diagonal elements of $\mathbf{W}$ are strictly positive. In particular, contrary to most work on kernel-based algorithms, our theoretical framework makes no assumptions regarding the positive semidefiniteness of the matrix (a symmetric matrix $\mathbf{W}$ is positive semidefinite if and only if for all vectors $\mathbf{u} \in \mathbb{R}^P$, $\mathbf{u}^\top \mathbf{W}\mathbf{u} \geqslant 0$). If in fact the matrix is positive semidefinite this can be exploited in the design of efficient approximation algorithms (see Section 4.4). But the spectral clustering algorithms presented in this chapter are not limited to positive semidefinite matrices. In particular, it differs from the usual assumption made in kernel methods. For similar clustering approaches based on kernel methods, see Filippone et al. (2008), Bie and Cristianini (2006), Bach and Harchaoui (2008).

A classical similarity matrix for clustering in $\mathbb{R}^d$ is the diagonally-scaled Gaussian similarity, defined between pairs of points $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d \times \mathbb{R}^d$ as:

$$\mathbf{W}(\mathbf{x}, \mathbf{y}) = \exp(-(\mathbf{x} - \mathbf{y})^\top \operatorname{Diag}(\boldsymbol{\alpha})(\mathbf{x} - \mathbf{y})),$$

where $\boldsymbol{\alpha} \in \mathbb{R}^d$ is a vector of positive parameters, and $\operatorname{Diag}(\boldsymbol{\alpha})$ denotes the $d \times d$ diagonal matrix with diagonal $\boldsymbol{\alpha}$. It is also very common to use such similarity matrices after transformation to a set of "features," where each feature can depend on the entire dataset $(\mathbf{x}_i)_{i=1,...,P}$ or a subset thereof (see, e.g., Shi and Malik, 2000, for an example from computational vision and see Section 5 of the current chapter for examples from speech separation). In Figure 1, we present a toy example in two dimensions with the Gaussian similarity.

In the context of *graph partitioning* where data points are vertices of an undirected graph and $\mathbf{W}_{ij}$ is defined to be one if there is an edge between i and j, and zero otherwise, $\mathbf{W}$ is often referred to as an "affinity matrix" (Chung, 1997).

2.2 Normalized cuts

We let $V = \{1, ..., P\}$ denote the index set of all data points. We wish to find R disjoint clusters, $A = (A_r)_{r \in \{1,...,R\}}$, where $\bigcup_r A_r = V$, that optimize a certain cost function. In this chapter, we consider the R-way normalized cut, $C(A, \mathbf{W})$, defined as follows (Shi and Malik, 2000, Gu et al., 2001). For two subsets A, B of V, define the total weight between A and B as $W(A, B) = \sum_{i \in A} \sum_{j \in B} \mathbf{W}_{ij}$. Then the normalized cut is equal to:

$$C(A, \mathbf{W}) = \sum_{r=1}^{R} \frac{W(A_r, V \backslash A_r)}{W(A_r, V)}. \tag{1}$$

Noting that $W(A_r, V) = W(A_r, A_r) + W(A_r, V \backslash A_r)$, we see that the normalized cut is small if for all r, the weight between the r-th cluster and the remaining data points is small

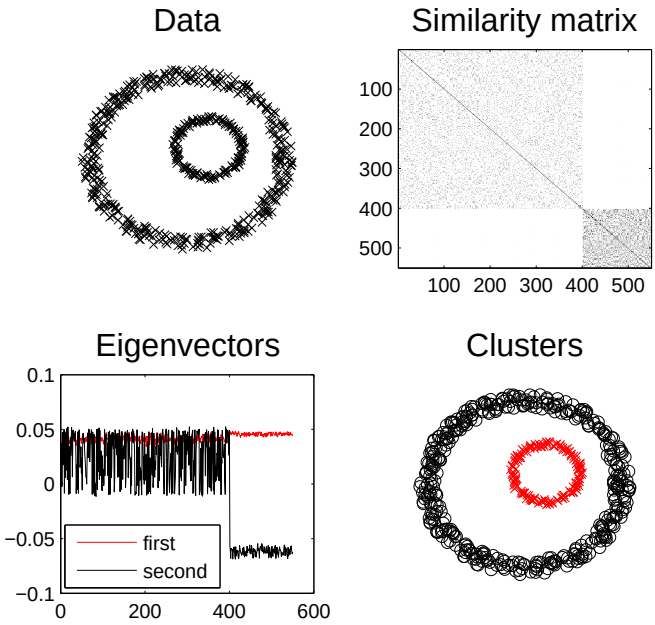

Figure 1: Toy examples in two dimensions.

compared to the weight within that cluster. The normalized cut criterion thus penalizes unbalanced partitions, while non-normalized criteria do not and often lead to trivial solutions (e.g., a cluster with only one point) when applied to clustering. In addition to being more immune to outliers, the normalized cut criterion and the ensuing spectral relaxations have a simpler theoretical asymptotic behavior when the number of data points tend to infinity (von Luxburg et al., 2005).

Let $\mathbf{e}_r$ be the indicator vector in $\mathbb{R}^P$ for the r-th cluster, i.e., $\mathbf{e}_r \in \{0,1\}^P$ is such that $\mathbf{e}_r$ has a nonzero component only for points in the r-th cluster. Knowledge of $\mathbf{E} = (\mathbf{e}_1, \ldots, \mathbf{e}_R) \in \mathbb{R}^{P \times R}$ is equivalent to knowledge of $A = (A_1, \ldots, A_R)$ and, when referring to partitions, we will use the two formulations interchangeably. A short calculation reveals that the normalized cut is then equal to:

$$C(E, \mathbf{W}) = \sum_{r=1}^{R} \frac{\mathbf{e}_r^\top (\mathbf{D} - \mathbf{W}) \mathbf{e}_r}{\mathbf{e}_r^\top \mathbf{D} \mathbf{e}_r}, \tag{2}$$

where $\mathbf{D}$ denotes the diagonal matrix whose i-th diagonal element is the sum of the elements in the i-th row of $\mathbf{W}$, i.e., $\mathbf{D} = \mathrm{Diag}(\mathbf{W}\mathbf{1})$, where $\mathbf{1}$ is defined as the vector in $\mathbb{R}^P$ composed of ones. Since we have assumed that all similarities are nonnegative, the matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$, usually referred to as the "Laplacian matrix," is a positive semidefinite matrix (Chung, 1997). In addition, its smallest eigenvalue is always zero, with eigenvector $\mathbf{1}$. Also, we have assumed that the diagonal of $\mathbf{W}$ is strictly positive, which implies that $\mathbf{D}$ is positive definite. Finally, in the next section, we also consider the normalized Laplacian matrix

defined as $\widetilde{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. This matrix is also positive definite with zero as its smallest eigenvalue, associated with eigenvector $\mathbf{D}^{1/2}\mathbf{1}$.

Minimizing the normalized cut is an NP-hard problem (Shi and Malik, 2000, Meila and Xu, 2003). Fortunately, tractable relaxations based on eigenvalue decomposition can be found.

2.3 Spectral relaxation

The following proposition, which extends a result of Shi and Malik (2000) for two clusters to an arbitrary number of clusters, gives an alternative description of the clustering task, and leads to a spectral relaxation:

Proposition 1 *For all partitions* $\mathbf{E}$ *into* R *clusters, the* R*-way normalized cut* $C(\mathbf{W}, \mathbf{E})$ *is equal to* $R - \operatorname{tr} \mathbf{Y}^\top \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}\mathbf{Y}$ *for any matrix* $\mathbf{Y} \in \mathbb{R}^{P \times R}$ *such that:*

(a) the columns of $\mathbf{D}^{-1/2}\mathbf{Y}$ *are piecewise constant with respect to the clusters* $\mathbf{E}$*,*

(b) $\mathbf{Y}$ *has orthonormal columns (*$\mathbf{Y}^\top\mathbf{Y} = \mathbf{I}$*).*

Proof The constraint (a) is equivalent to the existence of a matrix $\mathbf{\Lambda} \in \mathbb{R}^{R \times R}$ such that $\mathbf{D}^{-1/2}\mathbf{Y} = \mathbf{E}\mathbf{\Lambda}$. The constraint (b) is thus written as $\mathbf{I} = \mathbf{Y}^\top\mathbf{Y} = \mathbf{\Lambda}^\top\mathbf{E}^\top\mathbf{D}\mathbf{E}\mathbf{\Lambda}$. The matrix $\mathbf{E}^\top\mathbf{D}\mathbf{E}$ is diagonal, with elements $\mathbf{e}_r^\top\mathbf{D}\mathbf{e}_r$ and is thus positive and invertible. The $R \times R$ matrix $\mathbf{M} = (\mathbf{E}^\top\mathbf{D}\mathbf{E})^{1/2}\mathbf{\Lambda}$ satisfies $\mathbf{M}^\top\mathbf{M} = \mathbf{I}$, i.e., $\mathbf{M}$ is orthogonal, which implies $\mathbf{I} = \mathbf{M}\mathbf{M}^\top = (\mathbf{E}^\top\mathbf{D}\mathbf{E})^{1/2}\mathbf{\Lambda}\mathbf{\Lambda}^\top(\mathbf{E}^\top\mathbf{D}\mathbf{E})^{1/2}$.

This immediately implies that $\mathbf{\Lambda}\mathbf{\Lambda}^\top = (\mathbf{E}^\top\mathbf{D}\mathbf{E})^{-1}$. Thus we have:

$$\begin{aligned} R - \operatorname{tr} \mathbf{Y}^\top(\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2})\mathbf{Y} &= R - \operatorname{tr} \mathbf{\Lambda}^\top\mathbf{E}^\top\mathbf{D}^{1/2}(\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2})\mathbf{D}^{1/2}\mathbf{E}\mathbf{\Lambda} \\ &= R - \operatorname{tr} \mathbf{\Lambda}^\top\mathbf{E}^\top\mathbf{W}\mathbf{E}\mathbf{\Lambda} \\ &= R - \mathbf{E}^\top\mathbf{W}\mathbf{E}\mathbf{\Lambda}\mathbf{\Lambda}^\top = \operatorname{tr} \mathbf{E}^\top\mathbf{W}\mathbf{E}(\mathbf{E}^\top\mathbf{D}\mathbf{E})^{-1} \\ &= C(\mathbf{W}, \mathbf{E}), \end{aligned}$$

which completes the proof. ■

By removing the constraint (a), we obtain a relaxed optimization problem, whose solutions involve the eigenstructure of $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ and which leads to the classical lower bound on the optimal normalized cut (Zha et al., 2002, Chan et al., 1994). The following proposition gives the solution obtained from the spectral relaxation[1]:

Proposition 2 *The maximum of* $\operatorname{tr} \mathbf{Y}^\top\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}\mathbf{Y}$ *over matrices* $\mathbf{Y} \in \mathbb{R}^{P \times R}$ *such that* $\mathbf{Y}^\top\mathbf{Y} = \mathbf{I}$ *is the sum of the* R *largest eigenvalues of* $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$*. It is attained at all* $\mathbf{Y}$ *of the form* $\mathbf{Y} = \mathbf{U}\mathbf{B}_1$ *where* $\mathbf{U} \in \mathbb{R}^{P \times R}$ *is any orthonormal basis of the* R*-th principal subspace of* $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ *and* $\mathbf{B}_1$ *is an arbitrary orthogonal matrix in* $\mathbb{R}^{R \times R}$*.*

Proof Let $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. The proposition is equivalent to the classical variational characterization of the sum of the R largest eigenvalues $\lambda_1(\widetilde{\mathbf{W}}) \geqslant \cdots \geqslant \lambda_R(\widetilde{\mathbf{W}})$ of $\widetilde{\mathbf{W}}$—a result known as Ky Fan's theorem (Overton and Womersley, 1993):

$$\lambda_1(\widetilde{\mathbf{W}}) + \cdots + \lambda_R(\widetilde{\mathbf{W}}) = \max\{\operatorname{tr} \mathbf{Y}^\top\widetilde{\mathbf{W}}\mathbf{Y}, \mathbf{Y} \in \mathbb{R}^{P \times R}, \mathbf{Y}^\top\mathbf{Y} = \mathbf{I}\},$$

1. Tighter relaxations that exploit the nonnegativity of cluster indicators can be obtained (Xing and Jordan, 2003). These lead to convex relaxations, but their solution cannot be simply interpreted in terms of eigenvectors.

where the maximum is attained for all matrices $\mathbf{Y}$ of the form $\mathbf{Y} = \mathbf{U}\mathbf{B}_1$, where $\mathbf{U} \in \mathbb{R}^{P \times R}$ is any orthonormal basis of the R-th principal subspace of $\widetilde{\mathbf{W}}$ and $\mathbf{B}_1$ is an arbitrary orthogonal matrix in $\mathbb{R}^{R \times R}$. Note that the R-th principal subspace is uniquely defined if and only if $\lambda_R \neq \lambda_{R+1}$ (i.e., there is a positive eigengap). ■

The solutions found by this relaxation will not in general be piecewise constant, i.e., they will not in general satisfy constraint (a) in Proposition 1, and thus the relaxed solution has to be projected back to the constraint set defined by (a), an operation we refer to as "rounding," due to the similarity with the rounding performed after a linear programming relaxation of an integer programming problem (Bertsimas and Tsitsiklis, 1997).

2.4 Rounding

Our rounding procedure is based on the minimization of a metric between the relaxed solution and the entire set of discrete allowed solutions. Different metrics lead to different rounding schemes. In this section, we present two different metrics that take into account the known invariances of the problem.

Solutions of the relaxed problem are defined up to an orthogonal matrix, i.e., $\mathbf{Y}_{eig} = \mathbf{U}\mathbf{B}_1$, where $\mathbf{U} \in \mathbb{R}^{P \times R}$ is any orthonormal basis of the R-th principal subspace of $\mathbf{M}$ and $\mathbf{B}_1$ is an arbitrary orthogonal matrix. The set of matrices $\mathbf{Y}$ that correspond to a partition $\mathbf{E}$ and that satisfy constraints (a) and (b) are of the form $\mathbf{Y}_{part} = \mathbf{D}^{1/2}\mathbf{E}(\mathbf{E}^\top\mathbf{D}\mathbf{E})^{-1/2}\mathbf{B}_2$, where $\mathbf{B}_2$ is an arbitrary orthogonal matrix.

Since both matrices are defined up to an orthogonal matrix, it makes sense to compare the *subspaces* spanned by their columns. A common way to compare subspaces is to compare the orthogonal projection operators on those subspaces (Golub and Loan, 1996), that is, to compute the Frobenius norm between $\mathbf{Y}_{eig}\mathbf{Y}_{eig}^\top = \mathbf{U}\mathbf{U}^\top$ and the orthogonal projection operator $\mathbf{\Pi}_0(\mathbf{W}, \mathbf{E})$ on the subspace spanned by the columns of $\mathbf{D}^{1/2}\mathbf{E} = \mathbf{D}^{1/2}(\mathbf{e}_1, \ldots, \mathbf{e}_r)$, equal to:

$$\begin{aligned}
\mathbf{\Pi}_0(\mathbf{W}, \mathbf{E}) &= \mathbf{Y}_{part}\mathbf{Y}_{part}^\top \\
&= \mathbf{D}^{1/2}\mathbf{E}(\mathbf{E}^\top\mathbf{D}\mathbf{E})^{-1}\mathbf{E}^\top\mathbf{D}^{1/2} \\
&= \sum_r \frac{\mathbf{D}^{1/2}\mathbf{e}_r\mathbf{e}_r^\top\mathbf{D}^{1/2}}{\mathbf{e}_r^\top\mathbf{D}\mathbf{e}_r}.
\end{aligned}$$

We thus define the following cost function:

$$J_1(\mathbf{W}, \mathbf{E}) = \tfrac{1}{2}\left\|\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top - \mathbf{\Pi}_0(\mathbf{W}, \mathbf{E})\right\|_F^2. \tag{3}$$

Other cost functions could be derived using different metrics between linear subspaces, but as shown in Section 2.5, the Frobenius norm between orthogonal projections has the appealing feature that it leads to a weighted K-means algorithm.[2]

2. Another natural possibility followed by Yu and Shi (2003) is to compare directly $\mathbf{U}$ (or a normalized version thereof) with the indicator matrix $\mathbf{E}$, up to an orthogonal matrix, which then has to be estimated. This approach leads to an alternating minimization scheme similar to K-means.

Using the fact that both $\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top$ and $\mathbf{\Pi}_0(\mathbf{W},\mathbf{E})$ are orthogonal projection operators on linear subspaces of dimension R, we have:

$$\begin{aligned} J_1(\mathbf{W},\mathbf{E}) &= \frac{1}{2}\operatorname{tr}\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top + \frac{1}{2}\operatorname{tr}\mathbf{\Pi}_0(\mathbf{W},\mathbf{E})\mathbf{\Pi}_0(\mathbf{W},\mathbf{E})^\top - \operatorname{tr}\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top\mathbf{\Pi}_0(\mathbf{W},\mathbf{E}) \\ &= \frac{R}{2} + \frac{R}{2} - \operatorname{tr}\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top\mathbf{\Pi}_0(\mathbf{W},\mathbf{E}) \\ &= R - \sum_r \frac{\mathbf{e}_r^\top\mathbf{D}^{1/2}\mathbf{U}(\mathbf{W})\mathbf{U}(\mathbf{W})^\top\mathbf{D}^{1/2}\mathbf{e}_r}{\mathbf{e}_r^\top\mathbf{D}\mathbf{e}_r}. \end{aligned}$$

Note that if the similarity matrix $\mathbf{W}$ has rank equal to R, then our cost function $J_1(\mathbf{W},\mathbf{E})$ is exactly equal to the normalized cut $C(\mathbf{W},\mathbf{E})$.

Alternative normalization of eigenvectors By construction of the orthonormal basis $\mathbf{U}$ of the R-dimensional principal subspace of $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, the P R-dimensional rows $\mathbf{u}_1,\ldots,\mathbf{u}_P \in \mathbb{R}^R$ are already *globally* normalized, i.e., they satisfy $\mathbf{U}^\top\mathbf{U} = \sum_{i=1}^P \mathbf{u}_i\mathbf{u}_i^\top = \mathbf{I}$. Additional renormalization of those eigenvectors has proved worthwhile in clustering applications (Scott and Longuet-Higgins, 1990, Weiss, 1999, Ng et al., 2002), as can be seen in the idealized situation in which the similarity is zero between points that belong to different clusters and strictly positive between points in the same clusters. In this situation, the eigenvalue 1 has multiplicity R, and $\mathbf{D}^{1/2}\mathbf{E}$ is an orthonormal basis of the principal subspace. Thus, any basis $\mathbf{U}$ of the principal subspace has rows which are located on orthogonal *rays* in $\mathbb{R}^R$, where the distance from the i-th row $\mathbf{u}_i$ to the origin is simply $\mathbf{D}_{ii}^{1/2}$. By normalizing each row by the value $\mathbf{D}_{ii}^{1/2}$ or by its norm $\|\mathbf{u}_i\|$, the rows become orthonormal points in $\mathbb{R}^R$ (in the idealized situation) and thus are trivial to cluster. Ng et al. (2002) have shown that when the similarity matrix is "close" to this idealized situation, the properly normalized rows tightly cluster around an orthonormal basis.

Our cost function characterizes the ability of the matrix $\mathbf{W}$ to produce the partition $\mathbf{E}$ when using its eigenvectors. Minimizing with respect to $\mathbf{E}$ leads to new clustering algorithms that we now present. Minimizing with respect to the matrix $\mathbf{W}$ for a given partition $\mathbf{E}$ leads to algorithms for learning the similarity matrix, as we show in Section 3 and Section 4.

2.5 Spectral clustering algorithms

In this section, we provide a variational formulation of our cost function. Those variational formulations lead naturally to K-means and weighted K-means algorithms for minimizing this cost function with respect to the partition. While K-means is often used heuristically as a post-processor for spectral clustering (Ng et al., 2002, Meila and Shi, 2002), our approach provides a mathematical foundation for the use of K-means.

The following theorem, inspired by the spectral relaxation of K-means presented by Zha et al. (2002), shows that the cost function can be interpreted as a weighted distortion measure:

Theorem 3 *Let* $\mathbf{W}$ *be a similarity matrix and let* $\mathbf{U} = (\mathbf{u}_1,\ldots,\mathbf{u}_P)^\top$*, where* $\mathbf{u}_p \in \mathbb{R}^R$*, be an orthonormal basis of the* R*-th principal subspace of* $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$*, and* $d_p = \mathbf{D}_{pp}$ *for*

Input: Similarity matrix $\mathbf{W} \in \mathbb{R}^{P \times P}$.

Algorithm:

1. Compute first R eigenvectors $\mathbf{U}$ of $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ where $\mathbf{D} = \mathrm{diag}(\mathbf{W}\mathbf{1})$.
2. Let $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_P)^\top \in \mathbb{R}^{P \times R}$ and $d_p = \mathbf{D}_{pp}$.
3. Initialize partition A.
4. Weighted K-means: While partition A is not stationary,
   a. For all r, $\boldsymbol{\mu}_r = \sum_{p \in A_r} d_p^{1/2} \mathbf{u}_p / \sum_{p \in A_r} d_p$
   b. For all p, assign p to A_r where $r = \arg\min_{r'} \|\mathbf{u}_p d_p^{-1/2} - \boldsymbol{\mu}_{r'}\|$

Output: partition A, distortion measure $\sum_r \sum_{p \in A_r} d_p \|\mathbf{u}_p d_p^{-1/2} - \boldsymbol{\mu}_r\|^2$

Figure 2: Spectral clustering algorithm that minimizes $J_1(\mathbf{W}, \mathbf{E})$ with respect to $\mathbf{E}$ with weighted K-mean. See Section 2.5 for the initialization of the partition A.

all p. For any partition $\mathbf{E} \equiv A$, we have

$$J_1(\mathbf{W}, \mathbf{E}) = \min_{(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_R) \in \mathbb{R}^{R \times R}} \sum_r \sum_{p \in A_r} d_p \|\mathbf{u}_p d_p^{-1/2} - \boldsymbol{\mu}_r\|^2.$$

Proof Let $D(\boldsymbol{\mu}, A) = \sum_r \sum_{p \in A_r} d_p \|\mathbf{u}_p d_p^{-1/2} - \boldsymbol{\mu}_r\|^2$. Minimizing $D(\boldsymbol{\mu}, A)$ with respect to μ is a decoupled least-squares problem and we get:

$$\begin{aligned}
\min_{\boldsymbol{\mu}} \mathbf{D}(\boldsymbol{\mu}, A) &= \textstyle\sum_r \sum_{p \in A_r} \mathbf{u}_p^\top \mathbf{u}_p - \sum_r \| \sum_{p \in A_r} d_p^{1/2} \mathbf{u}_p \|^2 / \left(\sum_{p \in A_r} d_p \right) \\
&= \textstyle\sum_p \mathbf{u}_p^\top \mathbf{u}_p - \sum_r \sum_{p,p' \in A_r} d_p^{1/2} d_{p'}^{1/2} \mathbf{u}_p^\top \mathbf{u}_{p'} / \left(\mathbf{e}_r^\top \mathbf{D} \mathbf{e}_r \right) \\
&= R - \textstyle\sum_r \mathbf{e}_r^\top \mathbf{D}^{1/2} \mathbf{U} \mathbf{U}^\top \mathbf{D}^{1/2} \mathbf{e}_r / \left(\mathbf{e}_r^\top \mathbf{D} \mathbf{e}_r \right) = J_1(\mathbf{W}, \mathbf{E}).
\end{aligned}$$

■

This theorem has an immediate algorithmic implication—to minimize the cost function $J_1(\mathbf{W}, \mathbf{E})$ with respect to the partition $\mathbf{E}$, we can use a weighted K-means algorithm. The resulting algorithm is presented in Figure 2.

The rounding procedures that we propose in this chapter are similar to those in other spectral clustering algorithms (Ng et al., 2002, Yu and Shi, 2003). Empirically, all such rounding schemes usually lead to similar partitions. The main advantage of our procedure—which differs from the others in being derived from a cost function—it that it naturally leads to an algorithm for learning the similarity matrix from data, presented in Section 3.

Initialization The K-means algorithm can be interpreted as a coordinate descent algorithm and is thus subject to problems of local minima. Thus good initialization is crucial for the practical success of the algorithm in Figure 2.

A similarity matrix $\mathbf{W}$ is said *perfect with respect to a partition* $\mathbf{E}$ with R clusters if the cost function $J_1(\mathbf{W}, \mathbf{E})$ is exactly equal to zero. This is true in at least two potentially distinct situations: (1) when the matrix $\mathbf{W}$ is block-constant, where the block structure follows the partition $\mathbf{E}$, and, as seen earlier, (2) when the matrix $\mathbf{W}$ is such that the

similarity between points in different clusters is zero, while the similarity between points in the same clusters is strictly positive (Weiss, 1999, Ng et al., 2002).

In both situations, the R cluster centroids are orthogonal vectors, and Ng et al. (2002) have shown that when the similarity matrix is "close" to the second known type of perfect matrices, those centroids are close to orthogonal. This lead to the following natural initialization of the partition A for the K-means algorithm in Figure 2 (Ng et al., 2002): select a point $\mathbf{u}_p$ at random, and successively select $R-1$ points whose directions are most orthogonal to the previously chosen points; then assign each data point to the closest of the R chosen points.

2.6 Variational formulation for the normalized cut

In this section, we show that there is a variational formulation of the normalized cut similar to Theorem 3 for positive semidefinite similarity matrices, i.e., for matrices that can be factorized as $\mathbf{W}=\mathbf{G}\mathbf{G}^\top$ where $\mathbf{G}\in\mathbb{R}^{P\times M}$, where $M\leqslant P$. Indeed we have the following theorem, whose proof is almost identical to the proof of Theorem 3:

Theorem 4 *If $\mathbf{W}=\mathbf{G}\mathbf{G}^\top$ where $\mathbf{G}\in\mathbb{R}^{P\times M}$, then for any partition $\mathbf{E}$, we have:*

$$C(\mathbf{W},\mathbf{E})=\min_{(\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_R)\in\mathbb{R}^{R\times R}}\sum_r\sum_{p\in A_r}\mathbf{d}_p\|\mathbf{g}_p d_p^{-1}-\boldsymbol{\mu}_r\|^2+R-\operatorname{tr}\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}. \quad (4)$$

This theorem shows that for positive semidefinite matrices, the normalized cut problem is equivalent to the minimization of a weighted distortion measure. However, the dimensionality of the space involved in the distortion measure is equal to the rank of the similarity matrices, and thus can be very large (as large as the number of data points). Consequently, this theorem does not lead straightforwardly to an efficient algorithm for minimizing normalized cuts, since a weighted K-means algorithm in very high dimensions is subject to severe local minima problems (see, e.g., Meila and Heckerman, 2001). See Dhillon et al. (2004) for further algorithms based on the equivalence between normalized cuts and weighted K-means.

3. Cost functions for learning the similarity matrix

Given a similarity matrix $\mathbf{W}$, the steps of a spectral clustering algorithms are (1) normalization, (2) computation of eigenvalues, and (3) partitioning of the eigenvectors using (weighted) K-means to obtain a partition $\mathbf{E}$. In this section, we assume that the partition $\mathbf{E}$ is given, and we develop a theoretical framework and a set of algorithms for learning a similarity matrix $\mathbf{W}$.

It is important to note that if if we put no constraints on $\mathbf{W}$, then there is a trivial solution, namely any perfect similarity matrix with respect to the partition $\mathbf{E}$, in particular, any matrix that is block-constant with the appropriate blocks. For our problem to be meaningful, we thus must consider a setting in which there are several datasets to partition and we have a parametric form for the similarity matrix. The objective is to learn parameters that generalize to unseen datasets with a similar structure. We thus assume that the similarity matrix is a function of a vector variable $\boldsymbol{\alpha}\in\mathbb{R}^F$, and develop a method for learning $\boldsymbol{\alpha}$.

Given a distance between partitions, a naive algorithm would simply minimize the distance between the true partition $\mathbf{E}$ and the output of the spectral clustering algorithm. However, the K-means algorithm that is used to cluster eigenvectors is a non-continuous map and the naive cost function would be non-continuous and thus hard to optimize. In this section, we first show that the cost function we have presented is an upper bound on the naive cost function; this upper bound has better differentiability properties and is amenable to gradient-based optimization. The function that we obtain is a function of eigensubspaces and we provide numerical algorithms to efficiently minimize such functions in Section 3.3.

3.1 Distance between partitions

Let $\mathbf{E} = (\mathbf{e}_r)_{r=1,\dots,R}$ and $\mathbf{F} = (\mathbf{f}_s)_{s=1,\dots,S}$ be two partitions of P data points with R and S clusters, represented by the indicator matrices of sizes $P \times R$ and $P \times S$, respectively. We use the following distance between the two partitions (Hubert and Arabie, 1985):

$$\begin{aligned} d(\mathbf{E},\mathbf{F}) &= \frac{1}{\sqrt{2}} \left\| \mathbf{E}(\mathbf{E}^\top\mathbf{E})^{-1}\mathbf{E}^\top - \mathbf{F}(\mathbf{F}^\top\mathbf{F})^{-1}\mathbf{F}^\top \right\| \qquad (5) \\ &= \frac{1}{\sqrt{2}} \left\| \sum_r \frac{\mathbf{e}_r\mathbf{e}_r^\top}{\mathbf{e}_r^\top\mathbf{e}_r} - \sum_s \frac{\mathbf{f}_s\mathbf{f}_s^\top}{\mathbf{f}_s^\top\mathbf{f}_s} \right\|_F \\ &= \frac{1}{\sqrt{2}} \left(R + S - 2\sum_{r,s} \frac{(\mathbf{e}_r^\top\mathbf{f}_s)^2}{(\mathbf{e}_r^\top\mathbf{e}_r)(\mathbf{f}_s^\top\mathbf{f}_s)} \right)^{1/2}. \end{aligned}$$

The term $\mathbf{e}_r^\top\mathbf{f}_s$ simply counts the number of data points which belong to the r-th cluster of $\mathbf{E}$ and the s-th cluster of $\mathbf{F}$. The function $d(\mathbf{E},\mathbf{F})$ is a distance for partitions, i.e., it is nonnegative and symmetric, it is equal to zero if and only if the partitions are equal, and it satisfies the triangle inequality. Moreover, if $\mathbf{F}$ has S clusters and $\mathbf{E}$ has R clusters, we have $0 \leqslant d(\mathbf{E},\mathbf{F}) \leqslant (\frac{R+S}{2} - 1)^{1/2}$. In simulations, we compare partitions using the squared distance.

3.2 Cost functions as upper bounds

We let $\mathbf{E}_1(\mathbf{W})$ denote the clustering obtained by minimizing the cost function $J_1(\mathbf{W},\mathbf{E})$ with respect to $\mathbf{E}$. The following theorem shows that our cost function is an upper bound on the distance between a partition and the output of the spectral clustering algorithm:

Theorem 5 *Let* $\eta(\mathbf{W}) = \max_p \mathbf{D}_{pp} / \min_p \mathbf{D}_{pp} \geqslant 1$. *If* $\mathbf{E}_1(\mathbf{W}) = \arg\min_R J_1(\mathbf{W},\mathbf{E})$, *then for all partitions* $\mathbf{E}$, *we have:*

$$d(\mathbf{E},\mathbf{E}_1(\mathbf{W}))^2 \leqslant 4\eta(\mathbf{W}) J_1(\mathbf{W},\mathbf{E}). \qquad (6)$$

The previous theorem shows that minimizing our cost function is equivalent to minimizing an upper bound on the true cost function. This bound is tight at zero, consequently, if we are able to produce a similarity matrix $\mathbf{W}$ with small $J_1(\mathbf{W},\mathbf{E})$ cost, then the matrix will provably lead to a partition that is close to $\mathbf{E}$. Note that the bound in Eq. (6) contains a constant term dependent on $\mathbf{W}$ and the framework can be slightly modified to take this into account (Bach and Jordan, 2006). In Section 3.4, we compare our cost function to previously proposed cost functions.

3.3 Functions of eigensubspaces

Our cost function, as defined in Eq. (3), depends on the R-th principal eigensubspace, i.e., the subspace spanned by the first R eigenvectors, $\mathbf{U} \in \mathbb{R}^{P\times R}$, of $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. In this section, we review classical properties of eigensubspaces, and present optimization techniques to minimize functions of eigensubspaces. In this section, we focus mainly on the cost function $J_1(\mathbf{W}, \mathbf{E})$ which is defined in terms of the projections onto the principal subspace of $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. In this section, we first assume that all considered matrices are positive semidefinite, so that all eigenvalues are nonnegative, postponing the treatment of the general case to Section 3.3.4.

3.3.1 Properties of eigensubspaces

Let $\mathcal{M}_{P,R}$ be the set of symmetric matrices such that there is a positive gap between the R-th largest eigenvalue and the (R+1)-th largest eigenvalue. The set $\mathcal{M}_{P,R}$ is open (Magnus and Neudecker, 1999), and for any matrix in $\mathcal{M}_{P,R}$, the R-th principal subspace $E_R(\mathbf{M})$ is uniquely defined and the orthogonal projection $\mathbf{\Pi}_R(\mathbf{M})$ on that subspace is an unique identifier of that subspace. If $\mathbf{U}_R(\mathbf{M})$ is an orthonormal basis of eigenvectors associated with the R largest eigenvalues, we have $\mathbf{\Pi}_R(\mathbf{M}) = \mathbf{U}_R(\mathbf{M})\mathbf{U}_R(\mathbf{M})^\top$, and the value is independent of the choice of the basis $\mathbf{U}_R(\mathbf{M})$. Note that the R-th eigensubspace is well defined even if some eigenvalues larger than the R-th eigenvalue coalesce (in which case, the R eigenvectors are not well defined but the R-th principal eigensubspace is).

The computation of eigenvectors and eigenvalues is a well-studied problem in numerical linear algebra (see, e.g., Golub and Loan, 1996). The two classical iterative techniques to obtain a few eigenvalues of a symmetric matrix are the *orthogonal iterations* (a generalization of the power method for one eigenvalue) and the *Lanczös method*.

The method of orthogonal iterations starts with a random matrix $\mathbf{V}$ in $\mathbb{R}^{P\times R}$, successively multiplies $\mathbf{V}$ by the matrix $\mathbf{M}$ and orthonormalizes the result with the QR decomposition. For almost all $\mathbf{V}$, the orthogonal iterations converge to the principal eigensubspace, and the convergence is linear with rate $\lambda_{R+1}(\mathbf{M})/\lambda_R(\mathbf{M})$, where $\lambda_1(\mathbf{M}) \geqslant \cdots \geqslant \lambda_{R+1}(\mathbf{M})$ are the R+1 largest eigenvalues of $\mathbf{M}$. The complexity of performing q steps of the orthogonal iterations is qR times the complexity of the matrix-vector product with the matrix $\mathbf{M}$. If $\mathbf{M}$ has no special structure, the complexity is thus $O(qRP^2)$. As discussed in Section 4.4, if special structure is present in $\mathbf{M}$ it is possible to reduce this to linear in P. The number of steps to obtain a given precision depends directly on the multiplicative eigengap $\varepsilon_R(\mathbf{M}) = \lambda_{R+1}(\mathbf{M})/\lambda_R(\mathbf{M}) \leqslant 1$; indeed this number of iterations is $O\left(\frac{1}{1-\varepsilon_R(\mathbf{M})}\right)$.

The Lanczös method is also an iterative method, one which makes better use of the available information to obtain more rapid convergence. Indeed the number of iterations is only $O\left(\frac{1}{(1-\varepsilon_R(\mathbf{M}))^{1/2}}\right)$, i.e., the square root of the number of iterations for the orthogonal iterations (Golub and Loan, 1996). Note that it is usual to perform subspace iterations on more than the desired number of eigenvalues in order to improve convergence (Bathe and Wilson, 1976).

Finally, in our setting of learning the similarity matrix, we can speed up the eigenvalue computation by initializing the power or Lanczös method with the eigensubspace of previous iterations. Other techniques are also available that can provide a similar speed-up by

efficiently tracking the principal subspace of slowly varying matrices (Comon and Golub, 1990, Edelman et al., 1999).

3.3.2 Approximation of eigensubspace and its differential

When learning the similarity matrix, the cost function and its derivatives are computed many times and it is thus worthwhile to use an efficient approximation of the eigensubspace as well as its differential. A very natural solution is to stop the iterative methods for computing eigenvectors at a fixed iteration q. The following proposition shows that for the method of power iterations, for almost all starting matrices $\mathbf{V} \in \mathbb{R}^{P \times R}$, the projection obtained by early stopping is an infinitely differentiable function:

Proposition 6 *Let* $\mathbf{V} \in \mathbb{R}^{P \times R}$ *be such that* $\eta = \max_{\mathbf{u} \in E_R(\mathbf{M})^\perp, \mathbf{v} \in \mathrm{range}(\mathbf{V})} \cos(\mathbf{u}, \mathbf{v}) < 1$. *Then if we let* $\mathbf{V}_q(\mathbf{M})$ *denote the results of* q *orthogonal iterations, the function* $\mathbf{V}_q(\mathbf{M})\mathbf{V}_q(\mathbf{M})^\top$ *is infinitely differentiable in a neighborhood of* $\mathbf{M}$*, and we have:* $\|\mathbf{V}_q(\mathbf{M})\mathbf{V}_q(\mathbf{M})^\top - \mathbf{\Pi}_R(\mathbf{M})\|_2 \leqslant \frac{\eta}{(1-\eta^2)^{1/2}}(|\lambda_{R+1}(\mathbf{M})|/|\lambda_R(\mathbf{M})|)^q$.

Proof Golub and Loan (1996) show that for all q, $\mathbf{M}^q\mathbf{V}$ always has rank R. When only the projection on the column space is sought, the result of the orthogonal iterations does not depend on the chosen method of orthonormalization (usually the QR decomposition), and the final result is theoretically equivalent to orthonormalizing at the last iteration. Thus $\mathbf{V}_q(\mathbf{M})\mathbf{V}_q(\mathbf{M})^\top = \mathbf{M}^q\mathbf{V}(\mathbf{V}^\top\mathbf{M}^{2q}\mathbf{V})^{-1}\mathbf{V}^\top\mathbf{M}^q$. $\mathbf{V}_q(\mathbf{M})\mathbf{V}_q(\mathbf{M})^\top$ is C^∞ since matrix inversion and multiplication are C^∞. The bound is proved in Golub and Loan (1996) for the QR orthogonal iterations, and since the subspaces computed by the two methods are the same, the bound also holds here. The derivative can easily be computed using the chain rule. ■

Note that numerically taking powers of matrices without care can lead to disastrous results (Golub and Loan, 1996). By using successive QR iterations, the computations can be made stable and the same technique can be used for the computation of the derivatives.

3.3.3 Potentially hard eigenvalue problems

In most of the literature on spectral clustering, it is taken for granted that the eigenvalue problem is easy to solve. It turns out that in many situations, the (multiplicative) eigengap is very close to one, making the eigenvector computation difficult (examples are given in the following section).

When the eigengap is close to one, a large power is necessary for the orthogonal iterations to converge. In order to avoid those situations, we regularize the approximation of the cost function based on the orthogonal iterations by a term which is large when the matrix $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ is expected to have a small eigengap, and small otherwise. We use the function $n(\mathbf{W}) = \operatorname{tr}\mathbf{W}/\operatorname{tr}\mathbf{D}$, which is always between 0 and 1, and is equal to 1 when $\mathbf{W}$ is diagonal (and thus has no eigengap).

We thus use the cost function defined as follows. Let $\mathbf{V} \in \mathbb{R}^{P \times R}$ be defined as $\mathbf{D}^{1/2}\mathbf{F}$, where the r-th column of $\mathbf{F}$ is the indicator matrix of a random subset of the r-th cluster normalized by the number of points in that cluster. This definition of $\mathbf{W}$ ensures that when $\mathbf{W}$ is diagonal, the cost function is equal to $R - 1$, i.e., if the power iterations are likely not to converge, then the value is the maximum possible true value of the cost.

Let $\mathbf{B}(\mathbf{W})$ be an approximate orthonormal basis of the projections on the R-th principal subspace of $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, based on orthogonal iterations starting from $\mathbf{V}$.[3]

The cost function that we use to approximate $J_1(\mathbf{W}, \mathbf{E})$ is

$$F_1(\mathbf{W}, \mathbf{E}) = \frac{1}{2} \left\| \mathbf{B}(\mathbf{W})\mathbf{B}(\mathbf{W})^\top - \mathbf{\Pi}_0(\mathbf{W}, \mathbf{E}) \right\|_F^2 - \kappa \log(1 - n(\mathbf{W})). \tag{7}$$

3.3.4 Negative eigenvalues

The spectral relaxation in Proposition 2 involves the largest eigenvalues of the matrix $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. The vector $\mathbf{D}^{1/2}\mathbf{1}$ is an eigenvector with eigenvalue 1; since we have assumed that $\mathbf{W}$ is pointwise nonnegative, 1 is the largest eigenvalue of $\widetilde{\mathbf{W}}$. Given any symmetric matrices (not necessarily positive semidefinite) orthogonal iterations will converge to eigensubspaces corresponding to eigenvalues which have largest magnitude, and it may well be the case that some negative eigenvalues of $\widetilde{\mathbf{W}}$ have larger magnitude than the largest (positive) eigenvalues, thus preventing the orthogonal iterations from converging to the desired eigenvectors. When the matrix $\mathbf{W}$ is positive semidefinite this is not possible. However, in the general case, eigenvalues have to be shifted so that they are all nonnegative. This is done by adding a multiple of the identity matrix to the matrix $\widetilde{\mathbf{W}}$, which does not modify the eigenvectors but simply potentially changes the signs of the eigenvalues. In our context adding exactly the identity matrix is sufficient to make the matrix positive; indeed, when $\mathbf{W}$ is pointwise nonnegative, then both $\mathbf{D} + \mathbf{W}$ and $\mathbf{D} - \mathbf{W}$ are *diagonally dominant* with nonnegative diagonal entries, and are thus positive semidefinite (Golub and Loan, 1996), which implies that $-\mathbf{I} \preccurlyeq \widetilde{\mathbf{W}} \preccurlyeq \mathbf{I}$, and thus $\mathbf{I} + \widetilde{\mathbf{W}}$ is positive semidefinite.

3.4 Empirical comparisons between cost functions

In this section, we study the ability of the various cost functions we have proposed to track the gold standard error measure in Eq. (5) as we vary the parameter α in the similarity matrix $\mathbf{W}_{pp'} = \exp(-\alpha \|\mathbf{x}_p - \mathbf{x}_{p'}\|^2)$. We study the cost function $J_1(\mathbf{W}, \mathbf{E})$ as well as their approximations based on the power method presented in Section 3.3.2. We also present results for two existing approaches, one based on a Markov chain interpretation of spectral clustering (Meila and Shi, 2002) and one based on the alignment (Cristianini et al., 2002) of $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ and $\mathbf{\Pi}_0$. Our experiment is based on the simple clustering problem shown in Figure 3(a). This apparently simple toy example captures much of the core difficulty of spectral clustering—nonlinear separability and thinness/sparsity of clusters (any point has very few near neighbors belonging to the same cluster, so that the weighted graph is sparse). In particular, in Figure 3(b) we plot the eigengap of the similarity matrix as a function of α, noting that for all optimum values of α, this gap is very close to one, and thus the eigenvalue problem is hard to solve. Worse, for large values of α, the eigengap becomes so small that the eigensolver starts to diverge. It is thus essential to prevent our learning algorithm from yielding parameter settings that lead to a very small eigengap. In Figure 3(e), we plot our approximation of the cost function based on the power method, and we see that, even without the additional regularization presented in Section 3.3.3, our

3. The matrix $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ always has the same largest eigenvalue 1 with eigenvector $\mathbf{D}^{1/2}\mathbf{1}$ and we could consider instead the $(R-1)$th principal subspace of $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} - \mathbf{D}^{1/2}\mathbf{1}\mathbf{1}^\top\mathbf{D}^{1/2}/(\mathbf{1}^\top\mathbf{D}\mathbf{1})$.

approximate cost function avoids a very small eigengap. The regularization presented in Section 3.3.3 strengthens this behavior.

In Figure 3(c) and (d), we plot the four cost functions against the gold standard. The gold standard curve shows that the optimal α lies above 2.5 on a log scale, and as seen in Figure 3(c) and (e), the minima of the new cost function and its approximation lie among these values. As seen in Figure 3(d), on the other hand, the alignment and Markov-chain-based cost functions show a poor match to the gold standard, and yield minima far from the optimum.

The problem with the latter cost functions is that these functions essentially measure the distance between the similarity matrix $\mathbf{W}$ (or a normalized version of $\mathbf{W}$) and a matrix T which (after permutation) is block-diagonal with constant blocks. Spectral clustering does work with matrices which are close to block-constant; however, one of the strengths of spectral clustering is its ability to work effectively with similarity matrices which are not block-constant, and which may exhibit strong variations among each block.

Indeed, in examples such as that shown in Figure 3, the optimal similarity matrix is very far from being block-diagonal with constant blocks. Rather, given that data points that lie in the same ring are in general far apart, the blocks are very sparse—not constant and full. Methods that try to find constant blocks cannot find the optimal matrices in these cases. In the language of spectral graph partitioning, where we have a weighted graph with weights $\mathbf{W}$, each cluster is a connected but very sparse graph. The power $\mathbf{W}^q$ corresponds to the q-th power of the graph; i.e., the graph in which two vertices are linked by an edge if and only if they are linked by a path of length no more than q in the original graph. Thus taking powers can be interpreted as "thickening" the graph to make the clusters more apparent, while not changing the eigenstructure of the matrix (taking powers of symmetric matrices only changes the eigenvalues, not the eigenvectors). Note that other clustering approaches based on taking powers of similarity matrices have been studied by Tishby and Slonim (2001) and Szummer and Jaakkola (2002); these differ from our approach in which we only take powers to approximate the cost function used for learning the similarity matrix.

4. Algorithms for learning the similarity matrix

We now turn to the problem of learning the similarity matrix from data. We assume that we are given one or more sets of data for which the desired clustering is known. The goal is to design a "similarity map," that is, a mapping from datasets of elements in $\mathcal{X}$ to the space of symmetric matrices with nonnegative elements. In this chapter, we assume that this space is parameterized. In particular, we consider diagonally-scaled Gaussian kernel matrices (for which the parameters are the scales of each dimension), as well as more complex parameterized matrices for speech separation in Section 5. In general we assume that the similarity matrix is a function of a vector variable $\boldsymbol{\alpha} \in \mathbb{R}^F$. We also assume that the parameters are in one-to-one correspondence with the features; setting one of these parameters to zero is equivalent to ignoring the corresponding feature.

4.1 Learning algorithm

We assume that we are given several related datasets with known partitions and our objective is to learn parameters of similarity matrices adapted to the overall problem. This

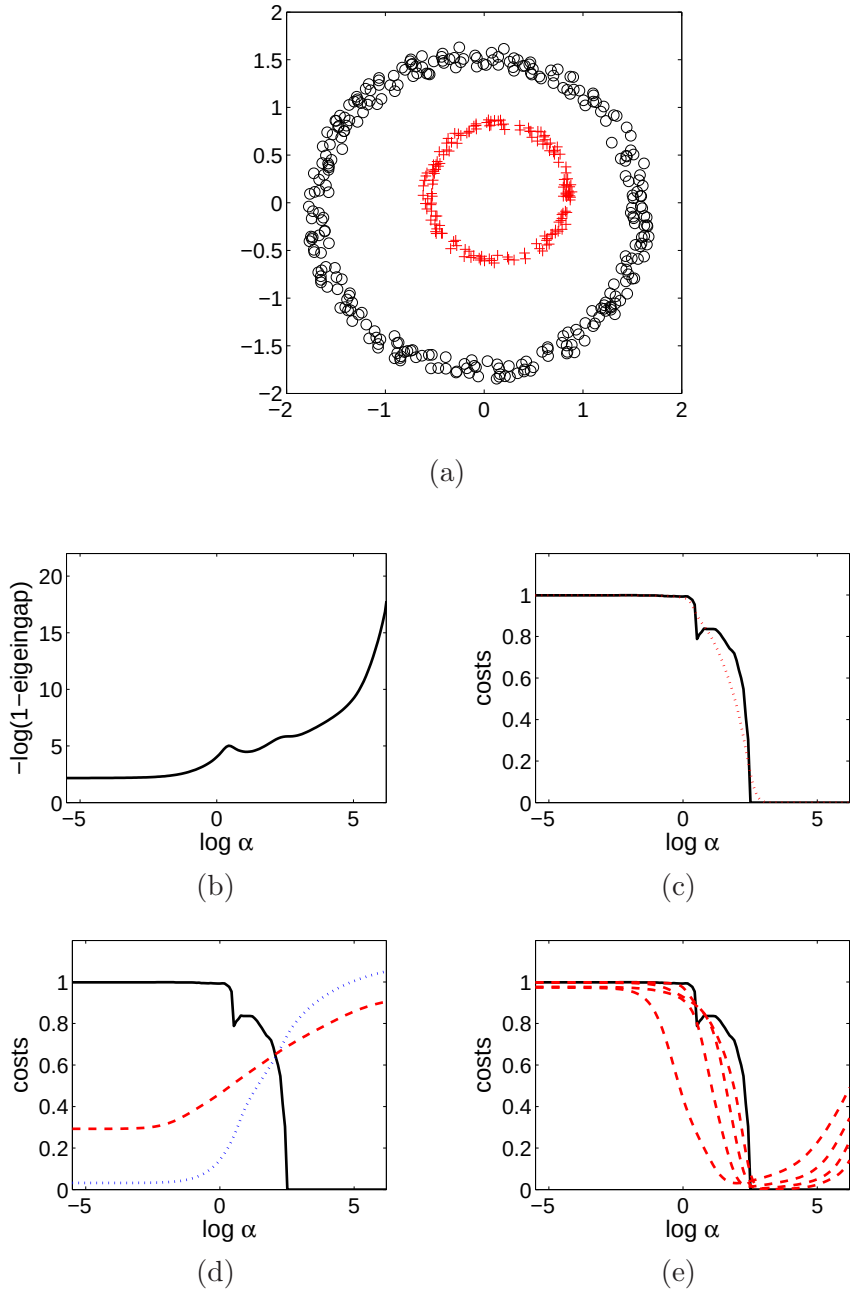

Figure 3: Empirical comparison of cost functions. (a) Data with two clusters (red crosses and blue circles). (b) Eigengap of the similarity matrix as a function of α. (c) Gold standard clustering error (black solid), spectral cost function J_1 (red dotted). (d) Gold standard clustering error (black solid), the alignment (red dashed), and a Markov-chain-based cost, divided by 20 (blue dotted). (e) Approximations based on the power method, with increasing power q: 2 4 16 32.

"supervised" setting is not uncommon in practice. In particular, as we show in Section 5, labeled datasets are readily obtained for the speech separation task by artificially combining separately-recorded samples. Note also that in the image segmentation domain, numerous images have been hand-labeled and a dataset of segmented natural images is available (Martin et al., 2001).

More precisely, we assume that we are given N datasets $\mathcal{D}_n$, $n \in \{1, \ldots, N\}$, of points in $\mathcal{X}$. Each dataset $\mathcal{D}_n$ is composed of P_n points $\mathbf{x}_{np}$, $p \in \{1, \ldots, P_n\}$. Each dataset is segmented; that is, for each n we know the partition $\mathbf{E}_n$. For each n and each α, we have a similarity matrix $\mathbf{W}_n(\boldsymbol{\alpha})$. The cost function that we use is $H(\boldsymbol{\alpha}) = \frac{1}{N} \sum_n F(\mathbf{W}_n(\boldsymbol{\alpha}), \mathbf{E}_n) + C \sum_{f=1}^{F} |\boldsymbol{\alpha}_f|$. The ℓ_1 penalty serves as a feature selection term, tending to make the solution sparse. The learning algorithm is the minimization of $H(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha} \in \mathbb{R}^F$, using the method of steepest descent.

Given that the complexity of the cost function increases with q, we start the minimization with small q and gradually increase q up to its maximum value. We have observed that for small q, the function to optimize is smoother and thus easier to optimize—in particular, the long plateaus of constant values are less pronounced. In some cases, we may end the optimization with a few steps of steepest descent using the cost function with the true eigenvectors, i.e., for $q = \infty$; this is particularly appropriate when the eigengaps of the optimal similarity matrices happen to be small.

4.2 Related work

Several other frameworks aim at learning the similarity matrices for spectral clustering or related procedures. Closest to our own work is the algorithm of Cour et al. (2005) which optimizes directly the eigenvectors of the similarity matrix, rather than the eigensubpaces, and is applied to image segmentation tasks. Although differently motivated, the frameworks of Meila and Shi (2002) and Shental et al. (2003) lead to similar convex optimization problems. The framework of Meila and Shi (2002) directly applies to spectral clustering, but we have shown in Section 3.4 that the cost function, although convex, may lead to similarity matrices that do not perform well. The probabilistic framework of Shental et al. (2003) is based on the model granular magnet of Blatt et al. (1997) and applies recent graphical model approximate inference techniques to solve the intractable inference required for the clustering task. Their framework leads to a convex maximum likelihood estimation problem for the similarity parameters, which is based on the same approximate inference algorithms. Among all those frameworks, ours has the advantage of providing theoretical bounds linking the cost function and the actual performance of spectral clustering.

4.3 Testing algorithm

The output of the learning algorithm is a vector $\boldsymbol{\alpha} \in \mathbb{R}^F$. In order to cluster previously unseen datasets, we compute the similarity matrix $\mathbf{W}$ and use the algorithm of Figure 2. In order to further enhance testing performance, we also adopt an idea due to Ng et al. (2002)—during testing, we vary the parameter $\boldsymbol{\alpha}$ along a direction $\boldsymbol{\beta}$. That is, for small λ we set the parameter value to $\boldsymbol{\alpha} + \lambda\boldsymbol{\beta}$ and perform spectral clustering, selecting λ such that the (weighted) distortion obtained after application of the spectral clustering algorithm of Figure 2 is minimal.

In our situation, there are two natural choices for the direction of search. The first is to use $\boldsymbol{\beta} = \boldsymbol{\alpha}/\|\boldsymbol{\alpha}\|$, i.e., we hold fixed the direction of the parameter but allow the norm to vary. This is natural for diagonally-scaled Gaussian kernel matrices. The second solution, which is more generally applicable, is to used the gradient of the individual cost functions, i.e., let $\mathbf{g}_n = \frac{dF(W_n(\boldsymbol{\alpha}),E_n)}{d\boldsymbol{\alpha}} \in \mathbb{R}^F$. If we neglect the effect of the regularization, at optimality, $\sum_n \mathbf{g}_n = 0$. We take the unit-norm direction such that $\sum_n (\boldsymbol{\beta}^\top \mathbf{g}_n)^2$ is maximum, which leads to choosing $\boldsymbol{\beta}$ as the largest eigenvector of $\sum_n \mathbf{g}_n \mathbf{g}_n^\top$.

4.4 Handling very large similarity matrices

In applications to vision and speech separation problems, the number of data points to cluster can be enormous: indeed, even a small 256×256 image leads to more than $P = 60,000$ pixels while three seconds of speech sampled at 5 kHz leads to more than $P = 15,000$ spectrogram samples. Thus, in such applications, the full matrix $\mathbf{W}$, of size $P \times P$, cannot be stored in main memory. In this section, we present approximation schemes for which the storage requirements are linear in P, for which the time complexity is linear in P, and which enable matrix-vector products to be computed in linear time. See Section 6.3 for an application of each of these methods to speech separation.

For an approximation scheme to be valid, we require the approximate matrix $\widetilde{\mathbf{W}}$ to be symmetric, with nonnegative elements, and with a strictly positive diagonal (to ensure in particular that D has a strictly positive diagonal). The first two techniques can be applied generally, while the last method is specific to situations in which there is natural one-dimensional structure, such as in speech or motion segmentation.

4.4.1 Sparsity

In applications to vision and related problems, most of the similarities are local, and most of the elements of the matrix $\mathbf{W}$ are equal to zero. If $Q \leqslant P(P+1)/2$ is the number of elements less than a given threshold (note that the matrix is symmetric so just the upper triangle needs to be stored), the storage requirement is linear in Q, as is the computational complexity of matrix-vector products. However, assessing which elements are equal to zero might take $O(P^2)$. Note that when the sparsity is low, i.e., when Q is large, using a sparse representation is unhelpful; only when the sparsity is expected to be high is it useful to consider such an option.

Thus, before attempting to compute all the significant elements (i.e., all elements greater than the threshold) of the matrix, we attempt to ensure that the resulting number of elements Q is small enough. We do so by selecting S random elements of the matrix and estimating from those S elements the proportion of significant elements, which immediately yields an estimate of Q.

If the estimated Q is small enough, we need to compute those Q numbers. However, although the total number of significant elements can be efficiently estimated, the indices of those significant elements cannot be obtained in less than $O(P^2)$ time without additional assumptions. A particular example is the case of diagonally-scaled Gaussian kernel matrices, for which the problem of computing all non-zero elements is equivalent to that of finding pairs of data points in an Euclidean space with distance smaller than a given threshold. We can exploit classical efficient algorithms to perform this task (Gray and Moore, 2001).

If $\mathbf{W}$ is an element-wise product of similarity matrices, only a subset of which have a nice structure, we can still use these techniques, albeit with the possibility of requiring more than Q elements of the similarity matrix to be computed.

4.4.2 Low-rank nonnegative decomposition

If the matrix $\mathbf{W}$ is not sparse, we can approximate it with a low-rank matrix. Following Fowlkes et al. (2001), it is computationally efficient to approximate each column of $\mathbf{W}$ by a linear combination of a set of randomly chosen columns: if I is the set of columns that are selected and J is the set of remaining columns, we approximate each column $\mathbf{w}_j$, $j \in J$, as a combination $\sum_{i \in I} \mathbf{H}_{ij} \mathbf{w}_i$. In the Nyström method of Fowlkes et al. (2001), the coefficient matrix $\mathbf{H}$ is chosen so that the squared error on the rows in indexed by I is minimum, i.e., $\mathbf{H}$ is chosen so that $\sum_{k \in I} \left(\mathbf{w}_j(k) - \sum_{i \in I} \mathbf{H}_{ij} \mathbf{w}_i(k) \right)^2$. Since $\mathbf{W}$ is symmetric, this only requires knowledge of the columns indexed by I. The solution of this convex quadratic optimization problem is simply $\mathbf{H} = \mathbf{W}(I, I)^{-1} \mathbf{W}(I, J)$, where for any sets A and B of distinct indices $\mathbf{W}(A, B)$ is the (A, B) block of $\mathbf{W}$. The resulting approximating matrix is symmetric and has a rank equal to the size of I.

When the matrix $\mathbf{W}$ is positive semidefinite, then the approximation remains positive semidefinite. However, when the matrix $\mathbf{W}$ is element-wise nonnegative, which is the main assumption in this chapter, then the approximation might not be and this may lead to numerical problems when applying the techniques presented in this chapter. In particular the approximated matrix $\mathbf{D}$ might not have a strictly positive diagonal. The following low-rank nonnegative decomposition has the advantage of retaining a pointwise nonnegative decomposition, while being only slightly slower. We use this decomposition in order to approximate the large similarity matrices, and the required rank is usually in the order of hundreds; this is to be contrasted with the approach of Ding et al. (2005), which consists in performing a nonnegative decomposition with very few factors in order to potentially obtain directly cluster indicators.

We first find the best approximation of $\mathbf{A} = \mathbf{W}(I, J)$ as $\mathbf{VH}$, where $\mathbf{V} = \mathbf{W}(I, I)$ and $\mathbf{H}$ is element-wise nonnegative. This can be done efficiently using algorithms for nonnegative matrix factorization (Lee and Seung, 2000). Indeed, starting from a random positive $\mathbf{H}$, we perform the following iteration until convergence:

$$\forall i, j, \ \mathbf{H}_{ij} \leftarrow \frac{\sum_k \mathbf{V}_{ki} \mathbf{A}_{kj} / (\mathbf{VH})_{kj}}{\sum_k \mathbf{V}_{ki}}. \tag{8}$$

The complexity of the iteration in Eq. (8) is $O(M^2 P)$, and empirically we usually find that we require a small number of iterations before reaching a sufficiently good solution. Note that the iteration yields a monotonic decrease in the following divergence:

$$D(\mathbf{A} | | \mathbf{VH}) = \sum_{ij} \left(\mathbf{A}_{ij} \log \frac{\mathbf{A}_{ij}}{(\mathbf{VH})_{ij}} - \mathbf{A}_{ij} + (\mathbf{VH})_{ij} \right).$$

We approximate $\mathbf{W}(J, J)$ by symmetrization,[4] i.e., $\mathbf{W}(J, I)\mathbf{H} + \mathbf{H}^\top \mathbf{W}(I, J)$. In order to obtain a better approximation, we ensure that the diagonal of $\mathbf{W}(J, J)$ is always used

4. For a direct low-rank symmetric nonnegative decomposition algorithm, see Ding et al. (2005).

Table 1: Performance on synthetic datasets: clustering errors (multiplied by 100) for method without learning (but with tuning) and for our learning method with and without tuning, with $N=1$ or 10 training datasets; D is the number of irrelevant features.

D	no learning	learning w/o tuning		learning with tuning	
		N=1	N=10	N=1	N=10
0	0	15.5	10.5	0	0
1	60.8	37.7	9.5	0	0
2	79.8	36.9	9.5	0	0
4	99.8	37.8	9.7	0.4	0
8	99.8	37	10.7	0	0
16	99.7	38.8	10.9	14	0
32	99.9	38.9	15.1	14.6	6.1

with its true (i.e., not approximated) value. Note that the matrices $\mathbf{H}$ found by nonnegative matrix factorization are usually sparse.

The storage requirement is $O(MP)$, where M is the number of selected columns. The complexity of the matrix-vector products is $O(MP)$. Empirically, the average overall complexity of obtaining the decomposition is $O(M^2P)$.

4.5 Simulations on toy examples

We performed simulations on synthetic datasets involving two-dimensional datasets similar to that shown in Figure 3, where there are two rings whose relative distance is constant across samples (but whose relative orientation has a random direction). We add D irrelevant dimensions of the same magnitude as the two relevant variables. The goal is thus to learn the diagonal scale $\boldsymbol{\alpha} \in \mathbb{R}^{D+2}$ of a Gaussian kernel that leads to the best clustering on unseen data. We learn $\boldsymbol{\alpha}$ from N sample datasets ($N=1$ or $N=10$), and compute the clustering error of our algorithm with and without adaptive tuning of the norm of $\boldsymbol{\alpha}$ during testing (cf. Section 4.3) on ten previously unseen datasets. We compare to an approach that does not use the training data: $\boldsymbol{\alpha}$ is taken to be the vector of all ones and we again search over the best possible norm during testing (we refer to this method as "no learning"). We report results in Table 1. Without feature selection, the performance of spectral clustering degrades very rapidly when the number of irrelevant features increases, while our learning approach is very robust, even with only one training dataset.

5. Speech separation as spectrogram segmentation

The problem of recovering signals from linear mixtures, with only partial knowledge of the mixing process and the signals—a problem often referred to as *blind source separation*—is a central problem in signal processing. It has applications in many fields, including speech processing, network tomography and biomedical imaging (Hyvärinen et al., 2001).

When the problem is over-determined, i.e., when there are no more signals to estimate (the sources) than signals that are observed (the sensors), generic assumptions such as statistical independence of the sources can be used in order to demix successfully (Hyvärinen et al., 2001). Many interesting applications, however, involve under-determined problems (more sources than sensors), where more specific assumptions must be made in order to demix. In problems involving at least two sensors, progress has been made by appealing to sparsity assumptions (Zibulevsky et al., 2002, Jourjine et al., 2000).

However, the most extreme case, in which there is only one sensor and two or more sources, is a much harder and still-open problem for complex signals such as speech. In this setting, simple generic statistical assumptions do not suffice. One approach to the problem involves a return to the spirit of classical engineering methods such as matched filters, and estimating specific models for specific sources—e.g., specific speakers in the case of speech (Roweis, 2001, Jang and Lee, 2003). While such an approach is reasonable, it departs significantly from the desideratum of "blindness." In this section we present an algorithm that is a blind separation algorithm—our algorithm separates speech mixtures from a single microphone without requiring models of specific speakers.

Our approach involves a "discriminative" approach to the problem of speech separation that is based on the spectral learning methodology presented in Section 4. That is, rather than building a complex model of speech, we instead focus directly on the task of separation and optimize parameters that determine separation performance. We work within a time-frequency representation (a spectrogram), and exploit the sparsity of speech signals in this representation. That is, although two speakers might speak simultaneously, there is relatively little overlap in the time-frequency plane if the speakers are different (Roweis, 2001, Jourjine et al., 2000). We thus formulate speech separation as a problem in segmentation in the time-frequency plane. In principle, we could appeal to classical segmentation methods from vision (see, e.g., Shi and Malik, 2000) to solve this two-dimensional segmentation problem. Speech segments are, however, very different from visual segments, reflecting very different underlying physics. Thus we must design features for segmenting speech from first principles.

5.1 Spectrogram

The spectrogram is a two-dimensional (time and frequency) redundant representation of a one-dimensional signal (Mallat, 1998). Let $\mathbf{f}[t], t = 0, \ldots, T-1$ be a signal in $\mathbb{R}^T$. The spectrogram is defined via windowed Fourier transforms and is commonly referred to as a short-time Fourier transform or as Gabor analysis (Mallat, 1998). The value $(\mathbf{U}f)_{mn}$ of the spectrogram at time window n and frequency m is defined as $(\mathbf{U}\mathbf{f})_{mn} = \frac{1}{\sqrt{M}} \sum_{t=0}^{T-1} \mathbf{f}[t]\mathbf{w}[t - na]e^{i2\pi mt/M}$, where $\mathbf{w}$ is a window of length T with small support of length c, and $M \geqslant c$. We assume that the number of samples T is an integer multiple of a and c. There are then $N = T/a$ different windows of length c. The spectrogram is thus an $N \times M$ image which provides a redundant time-frequency representation of time signals[5] (see Figure 4).

5. In our simulations, the sampling frequency is $f_0 = 5.5$ kHz and we use a Hanning window of length $c = 216$ (i.e., 43.2 ms). The spacing between window is equal to $a = 54$ (i.e., 10.8 ms). We use a 512-point FFT ($M = 512$). For a speech sample of length 4 seconds, we have $T = 22,000$ samples and then $N = 407$, which yields $\approx 2 \times 10^5$ spectrogram samples.

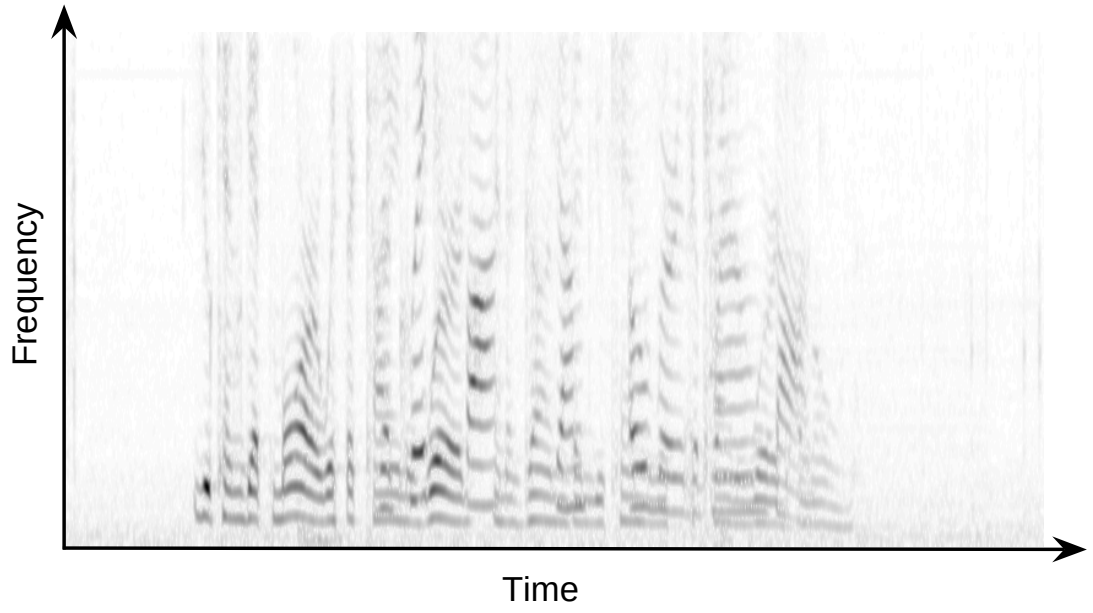

Figure 4: Spectrogram of speech (two simultaneous English speakers). The gray intensity is proportional to the amplitude of the spectrogram.

Inversion Our speech separation framework is based on the segmentation of the spectrogram of a signal $\mathbf{f}[t]$ in $R \geqslant 2$ disjoint subsets A_i, $i = 1, \ldots, R$ of $[0, N-1] \times [0, M-1]$. This leads to R spectrograms $\mathbf{U}_i$ such that $(\mathbf{U}_i)_{mn} = \mathbf{U}_{mn}$ if $(m, n) \in A_i$ and zero otherwise. We now need to find R speech signals $\mathbf{f}_i[t]$ such that each $\mathbf{U}_i$ is the spectrogram of $\mathbf{f}_i$. In general there are no exact solutions (because the representation is redundant), and a classical technique is to find the minimum ℓ_2 norm approximation, i.e., find $\mathbf{f}_i$ such that $\|\mathbf{U}_i - \mathbf{U}\mathbf{f}_i\|^2$ is minimal (Mallat, 1998). The solution of this minimization problem involves the pseudo-inverse of the linear operator $\mathbf{U}$ (Mallat, 1998) and is equal to $\mathbf{f}_i = (\mathbf{U}^*\mathbf{U})^{-1}\mathbf{U}^*\mathbf{U}_i$, where $\mathbf{U}^*$ is the (complex) adjoint of the linear operator $\mathbf{U}$. By our choice of window (Hanning), $\mathbf{U}^*\mathbf{U}$ is proportional to the identity matrix, so that the solution to this problem can simply be obtained by applying the adjoint operator $\mathbf{U}^*$. Other techniques for spectrogram inversion could be used (Griffin and Lim, 1984, Mallat, 1998, Achan et al., 2003)

5.2 Normalization and subsampling

There are several ways of normalizing a speech signal. In this chapter, we chose to rescale all speech signals as follows: for each time window n, we compute the total energy $e_n = \sum_m |\mathbf{U}\mathbf{f}_{mn}|^2$, and its 20-point moving average. The signals are normalized so that the 90th percentile of those values is equal to one.

In order to reduce the number of spectrogram samples to consider, for a given pre-normalized speech signal, we threshold coefficients whose magnitudes are less than a value that was chosen so that the resulting distortion is inaudible.

5.3 Generating training samples

Our approach is based on the learning algorithm presented in Section 4. The training examples that we provide to this algorithm are obtained by mixing separately-normalized speech signals. That is, given two volume-normalized speech signals, $\mathbf{f}_1$ and $\mathbf{f}_2$, of the same duration, with spectrograms $\mathbf{U}_1$ and $\mathbf{U}_2$, we build a training sample as $\mathbf{U}^{train} = \mathbf{U}_1 + \mathbf{U}_2$, with a segmentation given by $z = \arg\min\{\mathbf{U}_1, \mathbf{U}_2\}$. In order to obtain better training partitions (and in particular to be more robust to the choice of normalization), we also search over all $\alpha \in [0, 1]$ such that the ℓ_2 reconstruction error obtained from segmenting/reconstructing using $z = \arg\min\{\alpha\mathbf{U}_1, (1-\alpha)\mathbf{U}_2\}$ is minimized. An example of such a partition is shown in Figure 5 (top).

5.4 Features and grouping cues for speech separation

In this section we describe our approach to the design of features for the spectral segmentation. We base our design on classical cues suggested from studies of perceptual grouping (Cooke and Ellis, 2001). Our basic representation is a "feature map," a two-dimensional representation that has the same layout as the spectrogram. Each of these cues is associated with a specific time scale, which we refer to as "small" (less than 5 frames), "medium" (10 to 20 frames), and "large" (across all frames). (These scales will be of particular relevance to the design of numerical approximation methods in Section 6.3). Any given feature is not sufficient for separating by itself; rather, it is the combination of several features that makes our approach successful.

5.4.1 Non-harmonic cues

The following non-harmonic cues have counterparts in visual scenes and for these cues we are able to borrow from feature design techniques used in image segmentation (Shi and Malik, 2000).

- **Continuity** Two time-frequency points are likely to belong to the same segment if they are close in time or frequency; we thus use time and frequency directly as features. This cue acts at a small time scale.

- **Common fate cues** Elements that exhibit the same time variation are likely to belong to the same source. This takes several particular forms. The first is simply *common offset* and *common onset*. We thus build an offset map and an onset map, with elements that are zero when no variation occurs, and are large when there is a sharp decrease or increase (with respect to time) for that particular time-frequency point. The onset and offset maps are built using oriented energy filters as used in vision (with one vertical orientation). These are obtained by convolving the spectrogram with derivatives of Gaussian windows (Shi and Malik, 2000).

  Another form of the common fate cue is *frequency co-modulation*, the situation in which frequency components of a single source tend to move in sync. To capture this cue we simply use oriented filter outputs for a set of orientation angles (8 in our simulations). Those features act mainly at a medium time scale.

5.4.2 Harmonic cues

This is the major cue for voiced speech (Gold and Morgan, 1999, Brown and Cooke, 1994, Bregman, 1990), and it acts at all time scales (small, medium and large): voiced speech is locally periodic and the local period is usually referred to as the pitch.

- **Pitch estimation** In order to use harmonic information, we need to estimate potentially several pitches. We have developed a simple pattern matching framework for doing this that we present in Bach and Jordan (2006). If S pitches are sought, the output that we obtain from the pitch extractor is, for each time frame n, the S pitches $\omega_{n1}, \ldots, \omega_{nS}$, as well as the strength y_{nms} of the s-th pitch for each frequency m.

- **Timbre** The pitch extraction algorithm presented in Appendix C also outputs the spectral envelope of the signal (Gold and Morgan, 1999). This can be used to design an additional feature related to timbre which helps integrate information regarding speaker identification across time. Timbre can be loosely defined as the set of properties of a voiced speech signal once the pitch has been factored out (Bregman, 1990). We add the spectral envelope as a feature (reducing its dimensionality using principal component analysis).

5.4.3 Building feature maps from pitch information

We build a set of features from the pitch information. Given a time-frequency point (m, n), let $s(m,n) = \arg\max_s \frac{y_{nms}}{(\sum_{m'} y_{nm's})^{1/2}}$ denote the highest energy pitch, and define the features $\omega_{ns(m,n)}$, $y_{nms(m,n)}$, $\sum_{m'} y_{nm's(m,n)}$, $\frac{y_{nms(m,n)}}{\sum_{m'} y_{nm's(m,n)}}$ and $\frac{y_{nms(m,n)}}{(\sum_{m'} y_{nm's(m,n)}))^{1/2}}$. We use a partial normalization with the square root to avoid including very low energy signals, while allowing a significant difference between the local amplitude of the speakers.

Those features all come with some form of energy level and all features involving pitch values ω should take this energy into account when the similarity matrix is built in Section 6. Indeed, this value has no meaning when no energy in that pitch is present.

6. Spectral clustering for speech separation

Given the features described in the previous section, we now show how to build similarity matrices that can be used to define a spectral segmenter. In particular, our approach builds *parameterized* similarity matrices, and uses the learning algorithm presented in Section 4 to adjust these parameters.

6.1 Basis similarity matrices

We define a set of "basis similarity" matrices for each set of cues and features defined in Section 5.4. Those basis matrices are then combined as described in Section 6.2 and the weights of this combination are learned as shown in Section 4.

For non-harmonic features, we use a radial basis function to define affinities. Thus, if $\mathbf{f}_a$ is the value of the feature for data point a, we use a basis similarity matrix defined as $\mathbf{W}_{ab} = \exp(-\|\mathbf{f}_a - \mathbf{f}_b\|^2)$. For a harmonic feature, on the other hand, we need to take into

account the strength of the feature: if $\mathbf{f}_a$ is the value of the feature for data point a, with strength y_a, we use $\mathbf{W}_{ab} = \exp(-\min\{y_a, y_b\}\|\mathbf{f}_a - \mathbf{f}_b\|^2)$.

6.2 Combination of similarity matrices

Given m basis matrices, we use the following parameterization of $\mathbf{W}$: $\mathbf{W} = \sum_{k=1}^{K} \gamma_k \mathbf{W}_1^{\alpha_{j1}} \times \cdots \times \mathbf{W}_m^{\alpha_{jm}}$, where the products are taken pointwise. Intuitively, if we consider the values of similarity as soft boolean variables, taking the product of two similarity matrices is equivalent to considering the conjunction of two matrices, while taking the sum can be seen as their disjunction. For our application to speech separation, we consider a sum of $K = 2$ matrices. This has the advantage of allowing different approximation schemes for each of the time scales, an issue we address in the following section.

6.3 Approximations of similarity matrices

The similarity matrices that we consider are huge, of size at least $50,000 \times 50,000$. Thus a significant part of our effort has involved finding computationally efficient approximations of similarity matrices.

Let us assume that the time-frequency plane is vectorized by stacking one time frame after the other. In this representation, the time scale of a basis similarity matrix $\mathbf{W}$ exerts an effect on the degree of "bandedness" of $\mathbf{W}$. Recall that the matrix $\mathbf{W}$ is referred to as band-diagonal with bandwidth B, if for all i, j, $|i - j| \geqslant B \Rightarrow \mathbf{W}_{ij} = 0$. On a small time scale, $\mathbf{W}$ has a small bandwidth; for a medium time scale, the band is larger but still small compared to the total size of the matrix, while for large scale effects, the matrix $\mathbf{W}$ has no band structure. Note that the bandwidth B can be controlled by the coefficient of the radial basis function involving the time feature n.

For each of these three cases, we have designed a particular way of approximating the matrix, while ensuring that in each case the time and space requirements are *linear* in the number of time frames, and thus linear in the duration of the signal to demix.

- **Small scale** If the bandwidth B is very small, we use a simple direct sparse approximation. The complexity of such an approximation grows linearly in the number of time frames.

- **Medium and large scale** We use a low-rank approximation of the matrix $\mathbf{W}$, as presented in Section 4.4. For mid-range interactions, we need an approximation whose rank grows with time, but whose complexity does not grow quadratically with time (see Section 4.4), while for large scale interactions, the rank is held fixed.

6.4 Experiments

We have trained our segmenter using data from four different male and female speakers, with speech signals of duration 3 seconds. There were 15 parameters to estimate using our spectral learning algorithm. For testing, we use mixes from speakers which were different from those in the training set.

In Figure 5, for two English speakers from the testing set, we show an example of the segmentation that is obtained when the two speech signals are known in advance (top

	Bound	Clust	Pitch	Freq
English (SNR)	2.3%	6.9%	31.1%	33.4%
English (SNR_{dB})	16.4	11.6	5.1	4.8
French (SNR)	3.3%	15.8%	35.4%	40.7%
French (SNR_{dB})	14.8	8.0	4.5	3.9

Table 2: Comparison of signal-to-noise ratios.

panel), a segmentation that would be used for training our spectral clustering algorithm, and in the bottom panel, the segmentation that is output by our algorithm.

Although some components of the "black" speaker are missing, the segmentation performance is good enough to obtain audible signals of reasonable quality. The speech samples for these examples can be downloaded from `http://www.di.ens.fr/~fbach/speech/`. On this web site, there are several additional examples of speech separation, with various speakers, in French and in English. Similarly, we present segmentation results for French speakers in Figure 6. Note that the same parameters were used for both languages and that the two languages were present in the training set. An important point is that our method does not require knowing the speakers in advance in order to demix successfully; rather, it is only necessary that the two speakers have distinct pitches most of the time (another but less crucial condition is that one pitch is not too close to twice the other one).

A complete evaluation of the robustness of our approach is outside the scope of this chapter; however, for the two examples shown in Figure 5 and Figure 6, we can compare signal-to-noise ratios for various competing approaches. Given the true signal $\mathbf{s}$ (known in our simulation experiments) and an estimated signal $\hat{\mathbf{s}}$, the signal-to-noise ratio (SNR) is defined as $SNR = \frac{\|\mathbf{s}-\hat{\mathbf{s}}\|^2}{\|\mathbf{s}\|^2}$, and is often reported in decibels, as $SNR_{dB} = -10\log_{10}\frac{\|\mathbf{s}-\hat{\mathbf{s}}\|^2}{\|\mathbf{s}\|^2}$. In order to characterize demixing performance, we use the maximum of the signal-to-noise ratios between the two true signals and the estimated signals (potentially after having permuted the estimated signals). In Table 2, we compare our approach ("Clust"), with the demixing solution obtained from the segmentation that would serve for training purposes ("Bound") (this can be seen as an upper bound on the performance of our approach). We also performed two baseline experiments: (1) In order to show that the combination of features is indeed crucial for performance, we performed K-means clustering on the estimated pitch to separate the two signals ("Pitch"). (2) In order to show that a full time-frequency approach is needed, and not simply frequency-based filtering, we used Wiener filters computed from the true signals ("Freq"). Note that to compute the four SNRs, the "Pitch" and "Freq" methods need the true signals, while the two other methods ("Clust" and "Bound") are pure separating approaches.

From the results in Table 2, we see that pitch alone is not sufficient for successful demixing (see the third column in the table). This is presumably due in part to the fact that pitch is not the only information available for grouping in the frequency domain, and due in part to the fact that multi-pitch estimation is a hard problem and multi-pitch estimation procedures tend to lead to noisy estimates of pitch. We also see (the forth column in the table) that a simple frequency-based approach is not competitive. This is not surprising

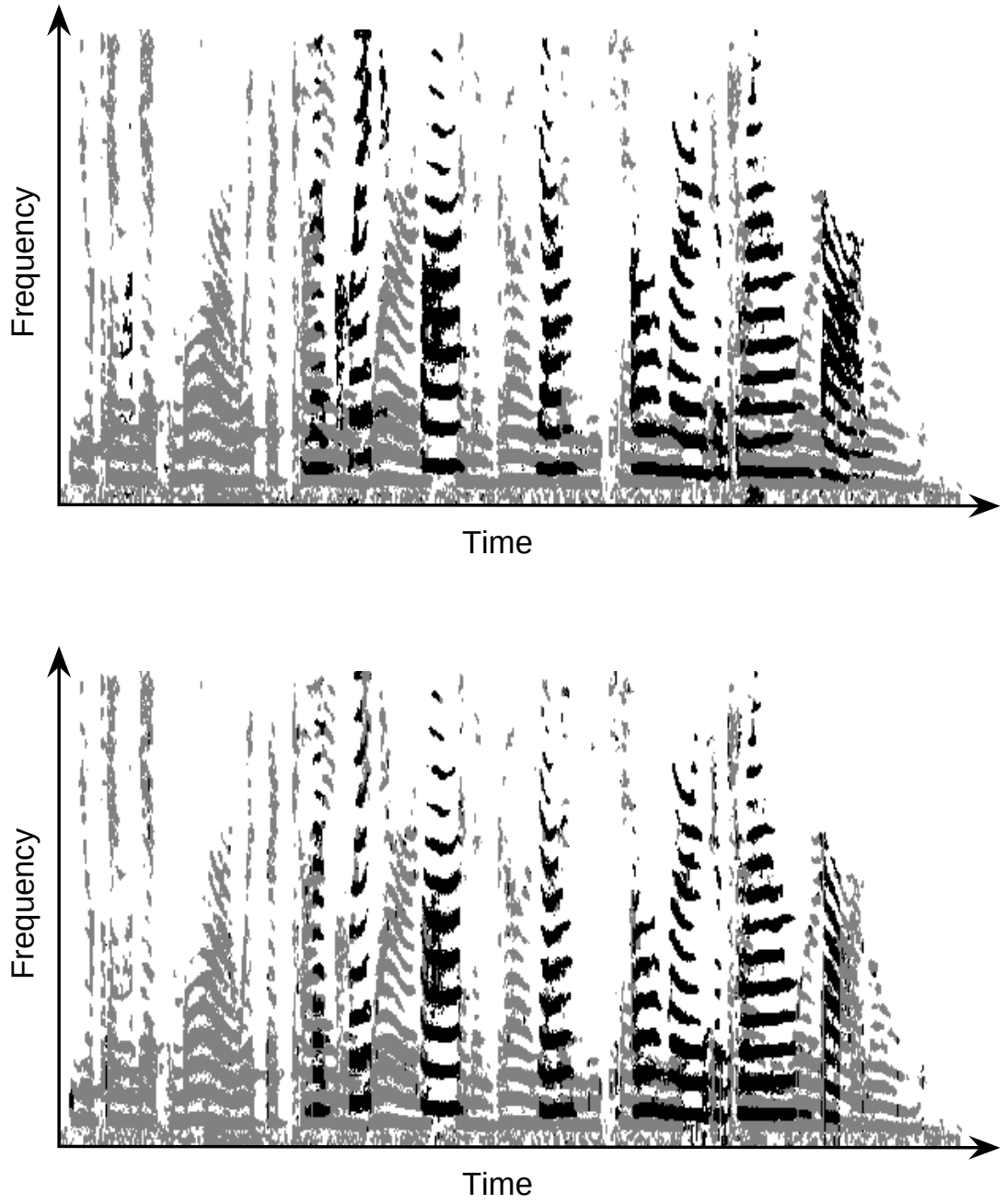

Figure 5: (Top) Optimal segmentation for the spectrogram of English speakers in Figure 4 (right), where the two speakers are "black" and "grey"; this segmentation is obtained from the known separated signals. (Bottom) The blind segmentation obtained with our algorithm.

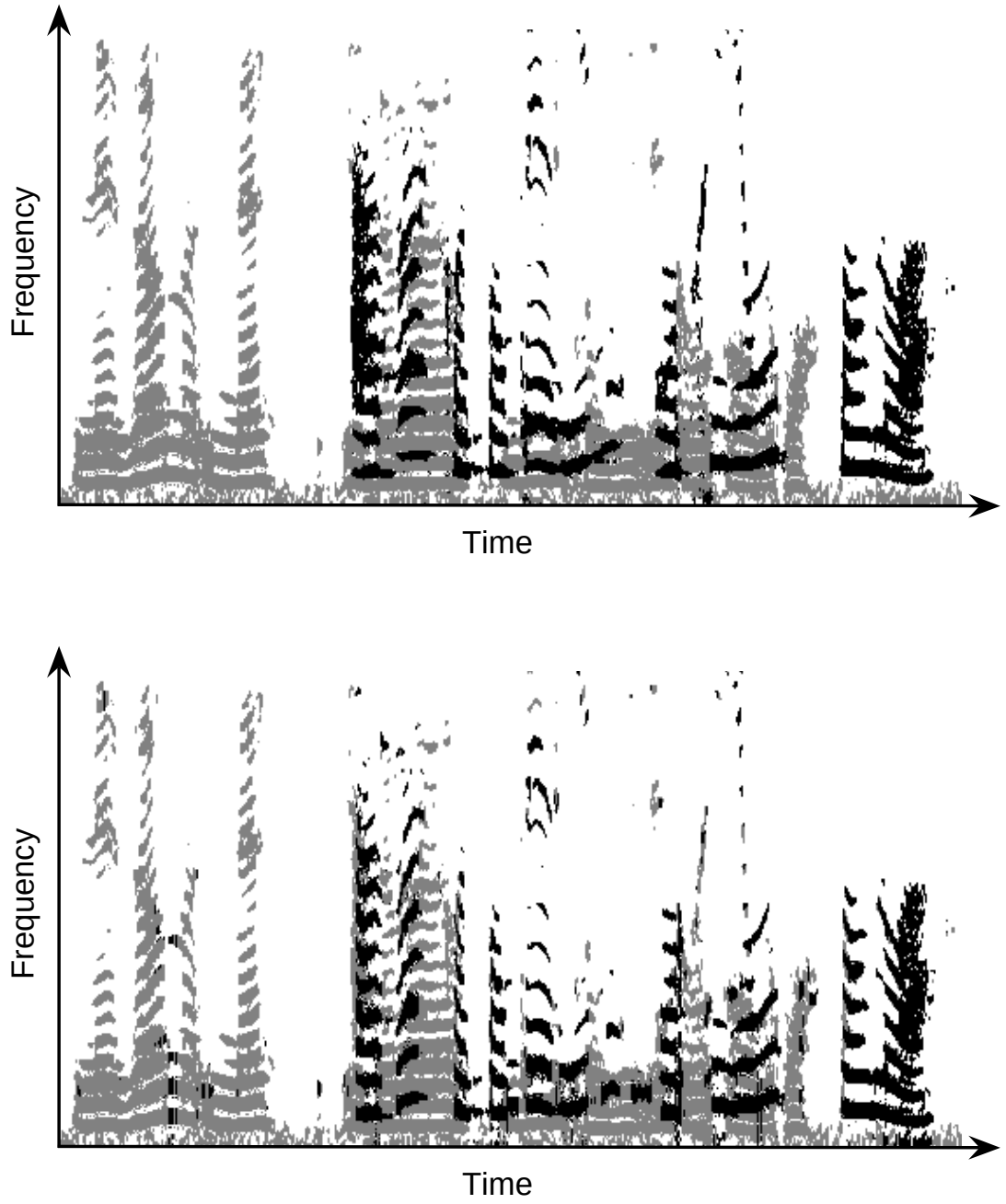

Figure 6: (Top) Optimal segmentation for the spectrogram of French speakers in Figure 4 (right), where the two speakers are "black" and "grey"; this segmentation is obtained from the known separated signals. (Bottom) The blind segmentation obtained with our algorithm.

because natural speech tends to occupy the whole spectrum (because of non-voiced portions and variations in pitch).

Finally, as mentioned earlier, there was a major computational challenge in applying spectral methods to single microphone speech separation. Using the techniques described in Section 6.3, the separation algorithm has linear running time complexity and memory requirement and, coded in Matlab and C, it takes 3 minutes to separate 4 seconds of speech on a 2 GHz processor with 1GB of RAM.

7. Conclusions

In this chapter, we have presented two sets of algorithms—one for spectral clustering and one for learning the similarity matrix. These algorithms can be derived as the minimization of a single cost function with respect to its two arguments. This cost function depends directly on the eigenstructure of the similarity matrix. We have shown that it can be approximated efficiently using the power method, yielding a method for learning similarity matrices that can cluster effectively in cases in which non-adaptive approaches fail. Note in particular that our new approach yields a spectral clustering method that is significantly more robust to irrelevant features than current methods.

We applied our learning framework to the problem of one-microphone blind source separation of speech. To do so, we have combined knowledge of physical and psychophysical properties of speech with learning algorithms. The former provide parameterized similarity matrices for spectral clustering, and the latter make use of our ability to generate segmented training data. The result is an optimized segmenter for spectrograms of speech mixtures. We have successfully demixed speech signals from two speakers using this approach.

Our work thus far has been limited to the setting of ideal acoustics and equal-strength mixing of two speakers. There are several obvious extensions that warrant investigation. First, the mixing conditions should be weakened and should allow some form of delay or echo. Second, there are multiple applications where speech has to be separated from non-stationary noise; we believe that our method can be extended to this situation. Third, our framework is based on segmentation of the spectrogram and, as such, distortions are inevitable since this is a "lossy" formulation (Jang and Lee, 2003, Jourjine et al., 2000). We are currently working on post-processing methods that remove some of those distortions. Finally, while the running time and memory requirements of our algorithm are linear in the duration of the signal to be separated, the resource requirements remain a concern. We are currently working on further numerical techniques that we believe will bring our method significantly closer to real-time.

References

K. Achan, S. Roweis, and B. Frey. Probabilistic inference of speech signals from phaseless spectrograms. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

F. Bach and Z. Harchaoui. Diffrac : a discriminative and flexible framework for clustering. In *Advances in Neural Information Processing Systems, 20*. MIT Press, 2008.

F. R. Bach and M. I. Jordan. Learning spectral clustering, with application to speech separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *International Conference on Machine Learning (ICML)*, 2003.

K.-J. Bathe and E. L. Wilson. *Numerical Methods in Finite Element Analysis*. Prentice Hall, 1976.

D. Bertsimas and J. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

T. De Bie and N. Cristianini. Fast sdp relaxations of graph cut clustering, transduction, and other combinatorial problems. *Journal of Machine Learning Research*, 7:1409–1436, 2006. ISSN 1533-7928.

M. Blatt, M. Wiesman, and E. Domany. Data clustering using a model granular magnet. *Neural Computation*, 9:1805–1842, 1997.

A. S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. MIT Press, 1990.

G. J. Brown and M. P. Cooke. Computational auditory scene analysis. *Computer Speech and Language*, 8:297–333, 1994.

P. K. Chan, M. D. F. Schlag, and J. Y. Zien. Spectral K-way ratio-cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, 13(9): 1088–1096, 1994.

F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

P. Comon and G. H. Golub. Tracking a few extreme singular values and vectors in signal processing. *Proceedings of the IEEE*, 78(8):1327–1343, 1990.

M. Cooke and D. P. W. Ellis. The auditory organization of speech and other sources in listeners and computational models. *Speech Communication*, 35(3-4):141–177, 2001.

T. Cour, N. Gogin, and J. Shi. Learning spectral graph segmentation. In *Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005.

N. Cristianini, J. Shawe-Taylor, and J. Kandola. Spectral kernel methods for clustering. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

I. Dhillon, Y. Guan, and B. Kulis. A unified view of kernel k-means, spectral clustering and and graph cuts. Technical Report #TR-04-25, University of Texas, Computer Science, 2004.

C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 2005.

A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1999.

M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 41(1):176–190, 2008. ISSN 0031-3203.

C. Fowlkes, S. Belongie, and J. Malik. Efficient spatiotemporal grouping using the Nyström method. In *IEEE conference on Computer Vision and Pattern Recognition (ECCV)*, 2001.

B. Gold and N. Morgan. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. Wiley Press, 1999.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.

A. G. Gray and A. W. Moore. N-Body problems in statistical learning. In *Advances in Neural Information Processing Systems, 13*. MIT Press, 2001.

D.W Griffin and J.S. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2):236–243, 1984.

M. Gu, H. Zha, C. Ding, X. He, and H. Simon. Spectral relaxation models and structure analysis for K-way graph clustering and bi-clustering. Technical report, Penn. State Univ, Computer Science and Engineering, 2001.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2001.

D. Higham and M. Kibble. A unified view of spectral clustering. Technical Report 02, University of Strathclyde, Department of Mathematics, 2004.

L. J. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, 2001.

G.-J. Jang and T.-W. Lee. A maximum likelihood approach to single-channel source separation. *Journal of Machine Learning Research*, 4:1365–1392, 2003.

A. Jourjine, S. Rickard, and O. Yilmaz. Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2000.

S. D. Kamvar, D. Klein, and C. D. Manning. Spectral learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems, 12*. MIT Press, 2000.

J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley, 1999.

S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1998.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision (ICCV)*, 2001.

M. Meila and D. Heckerman. An experimental comparison of several clustering and initialization methods. *Machine Learning*, 42(1):9–29, 2001.

M. Meila and J. Shi. Learning segmentation by random walks. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

M. Meila and L. Xu. Multiway cuts and spectral clustering. Technical report, University of Washington, Department of Statistics, 2003.

T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

M. L. Overton and R. S. Womersley. Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrics. *Mathematical Programming*, 62:321–357, 1993.

S. T. Roweis. One microphone source separation. In *Advances in Neural Information Processing Systems, 13*. MIT Press, 2001.

G.L. Scott and H. C. Longuet-Higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. In *British Machine Vision Conference*, 1990.

N. Shental, A. Zomet, T. Hertz, and Y. Weiss. Learning and inferring image segmentations using the GBP typical cut algorithm. In *International Conference on Computer Vision (ICCV)*, 2003.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

N. Tishby and N. Slonim. Data clustering by Markovian relaxation and the information bottleneck method. In *Advances in Neural Information Processing Systems, 13*. MIT Press, 2001.

U. von Luxburg, O. Bousquet, and M. Belkin. Limits of spectral clustering. In *Advances in Neural Information Processing Systems, 17*. MIT Press, 2005.

K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained K-means clustering with background knowledge. In *International Conference on Machine Learning (ICML)*, 2001.

Y. Weiss. Segmentation using eigenvectors: a unifying view. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 1999.

E. P. Xing and M. I. Jordan. On semidefinite relaxation for normalized k-cut and connections to spectral clustering. Technical Report UCB/CSD-03-1265, EECS Department, University of California, Berkeley, 2003.

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems, 15*. MIT Press, 2003.

S. X. Yu and J. Shi. Grouping with bias. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

S. X. Yu and J. Shi. Multiclass spectral clustering. In *International Conference on Computer Vision (ICCV)*, 2003.

H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for K-means clustering. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.

M. Zibulevsky, P. Kisilev, Y. Y. Zeevi, and B. A. Pearlmutter. Blind source separation via multinode sparse representation. In *Advances in Neural Information Processing Systems, 14*. MIT Press, 2002.