Analysis and Extension of Sparse Representations in Signal Classification

By

CHELSEA A. WEAVER

B.S. (University of Washington) 2009

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

MATHEMATICS

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Naoki Saito (Chair)

Thomas Strohmer

Jesus De Loera

Committee in Charge

2016

To Chris, for firmly telling me to stay the course. Consider the river crossed.

Contents

Chelsea A. Weaver
August 2016
Mathematics

Analysis and Extension of Sparse Representations in Signal Classification

Abstract

Classification, the task of assigning class labels to incoming data samples based on a training set, has as many applications as it has approaches. Our focus here is on classification algorithms that seek to represent the given test sample as an (often regularized) linear combination of the training samples. This dissertation has two parts:

First, we assess the representation-based approach to classification and the assumptions required for such methods to be effective. Wright et al.'s popular *sparse representation-based classification* (SRC) will be at the forefront of our analysis. By minimizing the ℓ^1-norm of the coefficient vector, SRC seeks a *sparse* decomposition of the test sample over the set of training samples, with classification to the most-contributing class. Though this algorithm is largely successful, it assumes that class manifolds are linear subspaces spanned by their training samples, which is often impractical. We proceed to modify SRC so that this restrictive assumption may be relaxed and the applicability of the algorithm extended. In particular, we use local principal component analysis to approximate the tangent hyperplane of the class manifold at each training sample, and we then use the basis vectors of these tangent hyperplanes to supplement a carefully-selected subset of training samples over which the test sample is sparsely decomposed. We show that the resulting, novel algorithm leads to better classification rates than SRC in cases of sparsely-sampled and/or nonlinear class manifolds, low noise, and stringent dimensionality reduction. Additionally, we state and prove a theorem that bounds the distance between the tangent vectors computed in our algorithm and the (ground truth) class manifolds. We also validate the use of sparsity (via ℓ^1-minimization) in the representation framework by comparing our modification of SRC to similar modifications of (i) ℓ^2-regularized (instead of ℓ^1-regularized) representation-based classification and (ii) a structured sparsity/block-sparse (minimization of the number of *classes* whose training samples have nonzero coefficients) approach to representation-based classification.

Second, we investigate the relationship between classification and sparsity, specifically, the two-fold question of whether sparsity in the representation is necessary for good classification performance, and whether or not the sparsest solution can be efficiently obtained in practice. Though under certain conditions, minimization of the ℓ^1-norm provably recovers the sparsest solution, we determine that the tractable approach to verifying this sparse recovery is fundamentally in conflict with the high coherence between training samples in the same class. Despite the lack of implementable recovery guarantee, we show experimentally that the sparsest solution is often recovered by ℓ^1-minimization in the case that the classes are well-separated. Further, through the use of a nonlinear transform designed so that sparse recovery conditions may be satisfied, we examine the relationship between the accuracy of SRC and the sparsity of its representation. We demonstrate that *approximate* (and not strict) equivalence between the ℓ^1-minimized solution and the sparsest solution is key to the success of SRC.

Acknowledgments

My research was conducted with government support under contract FA9550-11-C-0028 and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a, as well as by National Science Foundation VIGRE DMS-0636297 and NSF DMS-1418779. My adviser, Naoki Saito, was supported by ONR grants N00014-12-1-0177 and N00014-16-1-2255, as well as NSF DMS-1418779, during my time at UC Davis.

I would like to acknowledge the help of François Meyer and Daniel Kaslovsky in understanding parts of their research. They responded quickly and very kindly to my emails, and their help greatly surpassed my expectations and aided the tangent bound modification in Appendix A of this dissertation.

My sincere thanks to Naoki Saito for his meticulous feedback and supportive mentoring style, as well as for taking me on as his student before I had any real idea of what having an adviser meant. I'd also like to thank my committee members, Thomas Strohmer and Jesus De Loera—your encouragement and belief in me means more than I can say. Additionally, I could not have done this without my big-office peers Ryan Halabi and Jeff Irion, for whom I had more questions than answers, and Calina Copos, who literally made the filing of this dissertation possible.

Lastly, I'd like to profusely thank my family and friends for all their patience, encouragement, and support. They had unwavering faith in me every step of the way and gave me praise and forgiveness when I needed it most.

CHAPTER 1

Introduction

Classification is the task of assigning labels to test samples (samples whose classes are unknown) given the class information of a training set. Many classification algorithms have been proposed for various important applications, including detection of liver cancer from CT scans in medical image analysis [**76**], loan default prediction via credit scoring [**47**], earthquake detection [**6**], and human action recognition [**66**]. The classification of handwritten digits has been applied to the computer-automated reading of postal zip codes [**58**], and algorithms for face recognition, critical to applications from national security to social media, have been proposed [**17**, **71**, **104**]. As we will see, there are many criteria involved in assessing the goodness of a classification method, and many proposed classifiers offer an improvement in only some specific area(s). Along with the deluge of big data and recent advances in artificial intelligence, the difficulty of the task drives the seemingly unending need for better (faster, more accurate, more informative) classification algorithms.

In this dissertation, we focus on classification algorithms that seek to *express* the given test sample using the set of training samples. It follows that training samples that occur prominently in the expression (e.g., representation or approximation) are in some way deeply similar to the test sample, and so we can correctly infer the test sample's class from the class(es) of these significant training samples. For instance, when the test sample is expressed as a linear combination of the training samples, the (absolute values of the) coefficients can indicate the weight or degree of similarity between the test sample and each training sample. Thus we can pick out significant training samples by looking at their coefficients. We will refer to this as *representation-based classification*. This dissertation has two parts:

In Part 1, we assess the representation-based approach to classification and the assumptions required for such methods to be effective. In particular, we focus on algorithms that express the test sample as a regularized linear combination of the entire set of training samples. The popular algorithm *sparse representation-based classification* (SRC) [**104**] will be at the forefront of our

analysis. Proposed in 2009 by Wright et al., SRC was motivated by the recent boom in the use of sparse representation in signal processing. The catalyst of these advancements was the discovery that the sparsest representation of a signal using an over-complete set of vectors (often called a *dictionary*) could be found in many situations by minimizing the ℓ^1-norm of the representation coefficient vector [**29**]. Since the ℓ^1-minimization problem is convex, this gave rise to a tractable approach to obtaining the sparsest solution in many signal processing applications.

SRC applies this relationship between the minimum ℓ^1-norm and the sparsest solution to classification. The algorithm seeks the sparsest decomposition of a test sample over the dictionary of training samples via ℓ^1-minimization, with classification to the class whose corresponding portion of the representation approximates the test sample with least error. The method assumes that class manifolds are linear subspaces, so that the test sample can be represented using training samples in its ground truth class. Wright et al. [**104**] argue that this is precisely the sparsest decomposition of the test sample over the training set. They make the case that sparsity is critical to high-dimensional image classification and that, if properly harnessed, it can lead to impressive classification performance, even on highly-corrupted or occluded images. Further, good results can be achieved regardless of the choice of image features that are used for classification, provided that the number of retained features is large enough [**104**]. Though SRC was originally applied to face recognition, similar methods have been employed in clustering [**39**], dimensionality reduction [**73**], and texture and handwritten digit classification [**108**].

As we will see, the effectiveness of SRC and similar representation-based algorithms hinges on the lower-dimensional structure of each class manifold and the size and representativeness of the training set. When the data is poorly-sampled from nonlinear manifolds, it may be impossible to express test samples as approximate linear combinations of their same-class training samples.

We proceed to modify SRC to specifically address these issues. We call the resulting algorithm *local principal component analysis sparse representation-based classification* (LPCA-SRC). Our algorithm is designed to increase the training set via basis vectors of the hyperplanes approximately tangent to the (unknown) class manifolds. This provides the two-fold benefit of counter-balancing the potential sparse sampling of class manifolds (especially in the case that they are nonlinear) and helping to regain some information lost due to the dimensionality reduction procedure applied to

the data set as pre-processing. The dictionary of training samples in SRC is replaced with a set of vectors consisting of training samples that are close to the given test sample and their corresponding tangent hyperplane basis vectors, increasing the accuracy and locality of the approximation of the test sample in terms of its ground truth class. We demonstrate that this modified dictionary leads to classification accuracy exceeding that of traditional SRC and related methods on a synthetic database and three popular face databases.

Towards quantifying the efficacy of LPCA-SRC, we next use a result by Kaslovsky and Meyer [**57**] to bound the maximum distance between the tangent vectors computed in LPCA-SRC and the ground truth class manifolds. This analysis produces a potential tool for the user to gauge whether our modification of SRC can improve classification accuracy (over the original SRC) on a given data set, under the framework that the produced tangent vectors are newly-generated, potentially noisy training samples.

We conclude Part 1 by further validating the importance of the sparse-representation framework in LPCA-SRC (i.e., the "SRC" in "LPCA-SRC"). In particular, we similarly modify two other representation-based classification algorithms (a method based on ℓ^2-regularization and one based on structured/block-sparsity) using tangent vectors, and we show that our modification improves each algorithm but that ℓ^1-minimization is needed for good classification accuracy in several cases.

In Part 2, we investigate the role of sparsity in SRC, specifically, the two-fold question of (i) whether sparsity (achieved by minimizing the number of training samples with nonzero coefficients in the representation of the test sample) can improve classification performance, and (ii) whether or not this can be achieved in practice, i.e., whether ℓ^1-minimization produces the sparsest solution. The problem is that practically-implementable recovery conditions under which ℓ^1-minimization is guaranteed to find the sparsest solution require that the vectors in the dictionary be *incoherent*, or in some way "spread out" in space. These guarantees hold with high probability, for example, on dictionaries of vectors that are randomly-generated from certain probability distributions and dictionaries consisting of randomly-selected rows of the discrete Fourier transform matrix [**12**, **14**, **29**]. Obviously, unlike these examples, data samples in the same class are often *highly-correlated*. In fact, strong inner-class similarity generally makes the data *easier* to classify.

We show that the assumptions under which Wright et al. argue for the effectiveness of sparsity in SRC are in direct contradiction with these recovery guarantees, leading us to question the role of sparsity in the SRC (and more broadly, classification) framework. However, using a randomly-generated database designed to model facial images, we show that ℓ^1-minimization can still recover the sparsest solution on highly-correlated data, provided that the classes are sufficiently well-separated. Thus the lack of implementable equivalence *guarantee* should not automatically imply lack of *equivalence* in SRC, at least on certain databases.

Lastly, we investigate the feasibility and implementation of a transform that maximally spreads out the training samples in each class while still maintaining the data set's class structure. Though there are strict limitations on the design of such a transform, which we describe in detail in Chapter 12, we demonstrate that the higher-dimensional space can allow for the application of equivalence guarantees without negatively affecting the data set's ability to be classified. This makes it possible to examine the relationship between classification accuracy and the sparsity of the coefficient vector in SRC, and how close this is to the (provably) sparsest solution. We demonstrate that *approximate* (and not strict) equivalence between the ℓ^1-minimized solution and the sparsest solution is the key to the success of SRC.

The dissertation is organized as follows: Part 1 concerns the analysis of and our proposed improvement to representation-based classification. In Chapter 2, we review the problem of classification and several key ideas necessary to understanding and assessing classification algorithms. In Chapter 3, we discuss several existing representation-based algorithms and evaluate their theoretical implications and efficacy. Chapters 4 and 5 contain the presentation and empirical evaluation of our proposed classification algorithm LPCA-SRC, a modification (or extension) of the methods presented in Chapter 3. In Chapter 6, we prove a theorem related to the success of our proposed method, and in Chapter 7, we describe and appraise some closely-related alternatives to LPCA-SRC based on the same ideology. A brief conclusion of Part 1 is contained in Chapter 8. We note that Chapters 4-7 contain original contributions.

Part 2 analyzes the use of sparsity in classification, particularly in the sparse representation framework. In Chapter 9, we motivate and review sparsity recovery guarantees, and in Chapter 10, we assess their applicability to classification data. Chapter 11 presents empirical findings relating

sparse recovery and class-structured data, and in Chapter 12, we investigate the feasibility of a nonlinear data transform to force the aforementioned recovery guarantees to hold. These last three chapters contain original contributions. We conclude Part 2 in Chapter 13.

Parts 1 and 2 are followed by our final remarks, and Appendix A contains the proof of the tangent bound modification discussed in Chapter 6.

1.1. Notation

We assume that the input data is represented by vectors in $\mathbb{R}^m$, and we denote the number of classes by $L(< \infty)$. The training set is denoted by $X_{\text{tr}} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_{N_{\text{tr}}}] \in \mathbb{R}^{m \times N_{\text{tr}}}$, and we refer to a given test sample by $\boldsymbol{y} \in \mathbb{R}^m$.

The ℓ^1-norm of a vector $\boldsymbol{x} \in \mathbb{R}^m$ with coordinates x_i, $1 \leq i \leq m$, is defined as

$$\|\boldsymbol{x}\|_1 := \sum_{i=1}^{m} |x_i|,$$

and it should not be confused with the ℓ^2-norm, which is given by

$$\|\boldsymbol{x}\|_2 := \Big(\sum_{i=1}^{m} x_i^2 \Big)^{1/2}.$$

We use the notation

$$\|\boldsymbol{x}\|_0 := \#\{x_i \,|\, x_i \neq 0\}$$

to refer to the ℓ^0-"norm," i.e., the number of nonzero coordinates of the vector $\boldsymbol{x}$. Note that this is just a pseudonorm because it does not satisfy homogeneity.

1.2. Lists of Figures and Tables

Figures

Tables

Part 1

Improving Representation-Based Classification

CHAPTER 2

Classification

In this chapter, we describe the classification task and its inherent difficulties, and we define important terminology and concepts that will be used frequently in later chapters.

2.1. The Task

2.1.1. What is Classification? We, as humans, have an almost uncanny ability to recognize and classify objects around us. After making a new acquaintance, we are able to recognize her the next time we see her. We can tell the difference between an image of a building and an image of a car and identify which is which, even when only part of the object is visible or when the image has very low-resolution. For the most part, we can read each other's handwriting, even though there are countless ways to form each letter and everyone does so slightly differently.

With the rise of big data, it has become increasingly important that we find methods of passing on our natural recognition ability to machines. Having access to large amounts of data is worthless if we cannot efficiently sort it. A crime scene technician does not have time to look at millions of police photographs one-by-one in order to identify a suspect from an image pulled off a street camera. Google does not house a room full of people who manually comb through images to give us our image search results. These processes must happen automatically and incredibly quickly, and ideally, return highly-accurate results that are not subject to human error.

This task is also important because many technological advances are contingent on the automated ability to classify. A self-driven car must be able to tell the difference between pedestrians and other cars (moving or stationary) in order to avoid or minimize damage in accidents. Doctors need the technology to detect malignant tumors from MRI scans, and it is important that these are not confused with benign tumors or noise in the image. At their core, spam email filters and computer virus detectors are exactly automated classifiers, and there are countless other examples.

Despite its relevance, automated classification is often very difficult. Consider, for example, that for face recognition, not only must the algorithm be able to handle within-class variation in properties such as expression, face/head angle, and changes in hair or makeup, it must also be robust to differences that may occur in the image setting, most notably, the lighting conditions [**71**]. Further, in real-world settings, the data may suffer from noise corruption.

Let us formalize the classification problem. We say that a set of samples has *class structure* if the samples can be sorted into *classes* wherein each sample shares certain properties with all other samples in its class; further, these common properties defining the class are in some way fundamentally different than those defining other classes. In face recognition, for example, the classes correspond to the different subjects (people) in the database. For identification purposes, each class is denoted by a unique label, and every sample in the class inherits that label. Given a data set with class structure, *classification* is the task of assigning labels to samples based on a small set of (already) labeled data. We call the set of samples whose class labels are known the *training set* and any sample we want to classify a *test* sample. A *classification algorithm* is a list of programmable instructions to perform classification.

2.1.2. Dependency on the Training Set. In this setup, it is easy to see that the quality of the training data critically affects the performance of the classification algorithm. For example, if we give a classification algorithm several handwritten samples of the digits $0, 1, \ldots, 9$ along with their manually-assigned labels, correctness of the algorithm's characterization of a new digit depends on the quantity and variety of these training samples. For instance, we would not expect the algorithm to classify a "2" correctly if there were only a few "2's" in the training set, especially if those "2's" were written very differently from our test sample. Ideally, we would supply the algorithm with a training set consisting of several samples of each digit written in a wide variety of handwriting styles.

Observe that computers are immediately at a disadvantage in the classification scenario: as humans, we have a large background of life experiences that we can draw on when making classification decisions, whereas algorithms must singularly rely on their training sets. Returning to the example of separating building and car images, an algorithm will only be able to recognize a small section of tire from a test image if there is a (portion of a) tire in a training image. In contrast,

humans have huge, seemingly infinite training sets to use in our classification decisions. How many cars have we seen in our lives? How many digits?

It might seem that the obvious solution is to ensure that our classification algorithm has access to a vast and diverse training set. And this is certainly the case in some applications in which training samples can be obtained cheaply. However, in many situations, it is prohibitively expensive or even impossible to obtain a large training set. Without an automated classification algorithm, the labeling of training samples must be performed by hand (i.e., by a human), and this can be extremely time-consuming. Consider, for example, cataloging hundreds of digits one-by-one. Further, in applications such as cancer detection, we might have only a handful of positive cases, i.e., MRI scans corresponding to subjects who were later treated for especially rare types of cancer. Because of these reasons, it is crucial that a widely-applicable classification algorithm be able to "make the most" of the training set, determining correct class assignments given only a limited amount of training data.

REMARK 2.1.1. Semi-supervised learning *is a general method of classification that uses information contained in* unlabeled *data (separate from the given test sample), in addition to the labeled/training data, in order to make classification decisions. This approach can be very effective in ameliorating problems resulting from the infeasibility of obtaining a large number of labeled samples [**119**], especially since unlabeled data is often cheaper and/or more available than labeled data. However, semi-supervised learning has its own disadvantages, e.g., the addition of a large amount of unlabeled data (generally required for this approach to be effective) may significantly increase the computational cost of performing the actual classification. Though we do not consider the semi-supervised learning approach in this dissertation, instead adhering to what is known as purely* supervised learning*, much of our work could potentially be modified for application in the semi-supervised framework. See, for example, the use of sparse-representation in semi-supervised learning via Cheng et al.'s* ℓ^1-graph *[**21**].*

2.1.3. Assessing Classification Algorithms. In assessing the goodness of a classification algorithm, we consider several things. Clearly, we want the algorithm to provide us with accurate results, i.e., we want a high percentage of its classification decisions to be correct. In many

applications, we often need the algorithm to maintain high accuracy in the presence of significant within-class variation and noise or other data corruption. Secondly, it must be computationally efficient, so that it can be applied to large (either in number of samples or sample dimension) data sets. A probably less obvious criterion is the *simplicity* of the algorithm, in both its implementation and comprehensibility. In order for a classification algorithm to be broadly adopted, it must be able to be understood by those outside the author's field. Lastly, we consider the *interpretability* of the classification results: does the algorithm provide us with information beyond the class assignment, such as which elements of the test sample are most important to determining its class, or the specific training samples that are most similar to it?

For illustration, consider the simplest method of classification: k-nearest neighbors (kNN) [**23**]. In kNN, the k closest neighbors of the given test sample (according to some distance metric) are selected from the training set, and the test sample is classified according to the most-represented class among these neighbors. When $k = 1$, the test sample is assigned to the class of its nearest neighbor in the training set. kNN can have surprisingly high classification accuracy when the distance metric used accurately captures class structure or when the data is properly pre-processed (see Section 2.2); however, it is generally quite sensitive to noise. The computational efficiency of kNN depends on the efficiency of computing the distance metric; when Euclidean distance is used, for example, it is relatively fast. Further, as we mentioned, it cannot be beat in terms of simplicity. However, kNN gives us no additional information beyond the test sample's class assignment.

More sophisticated classification methods include naive Bayes classification, artificial neural networks, linear discriminant analysis [**42**, **74**], and support vector machines [**22**], each having their own strengths and weaknesses. Due to the difficulty of the classification task, the development of algorithms that rate highly against all the above criteria is still very much an open problem.

As mentioned in the introduction, in this dissertation we focus on the *representation-based approach to classification*, which is described in detail in Chapter 3. Though this type of algorithm can be applied to digit/object classification and many of the other examples discussed in Section 2.1.1 above (recall, for instance, the various extensions of SRC mentioned in Chapter 1), it is especially well-suited for *face recognition*, and so this application will play a recurring role throughout our work. In particular, we propose a new representation-based classification algorithm in Chapter

4, and we show in Chapters 5, 7, and 8 that it performs better against some of the above criteria than existing similar methods.

2.2. Pre-Processing and Feature Extraction

Before data can be classified, it must be formatted in a way that can be read by the algorithm. Though this depends on the algorithm used, we will assume that the input data is given to us as vectors, as is common. For example, if we wish to classify a set of images, we will prepare the data by taking each image and concatenating the transposed rows of pixel values into one long vector. Thus if the image is originally $p \times p$ pixels, the resulting vector will have length p^2.

There are numerous operations on the (vectorized) data that can be performed prior to classification in order to improve the performance of the algorithm. Essentially, these processes eliminate differences in the samples that are not relevant to classification. For example, if the data samples are viewed as realizations of a random variable, *sphering* or *whitening* is the process of transforming the data so that the sample covariance matrix of the transformed data is equal to the identity matrix. This tells the algorithm that the covariance between samples should not be considered when determining class. Alternatively, *centering* the data shifts each vector by the mean vector of the data set, so that the resulting data is centered around the origin; this should be used when the algorithm automatically considers the samples' positions around $\mathbf{0}$ and we want it to be ignored. Algorithms that are sensitive to differences in the norms of the data samples or the scaling of their coordinates may benefit from sample or coordinate *normalization* when these attributes are not indicative of the samples' class structure. More complicated pre-processing methods include techniques to reduce sample noise [**61**] and to fill in missing values [**75**], e.g., those lost due to sample degradation.

Another procedure that is extremely effective in improving both the accuracy and efficiency of the classification algorithm is *feature extraction*. The simplest feature extraction methods are designed to simply pick out the coordinates of the samples that are most relevant (with respect to some criterion) to determining class. This is called *feature selection*. The irrelevant coordinates

are determined using the training data and removed, thus helping to ensure that the subsequently-applied classification algorithm considers only information important to the task at hand. Further, by making the data samples more concise (shorter), the algorithm will run more efficiently.

More sophisticated feature extraction methods involve actually computing new features from the data samples and performing classification on these feature vectors. We describe some examples:

- *Dimensionality reduction:* Methods to perform dimensionality reduction are based on the assumption that the data lies in some lower-dimensional space, i.e., that samples originally in $\mathbb{R}^D$ can be expressed as vectors in $\mathbb{R}^m$, $m < D$, after a transformation to a new coordinate system. In *principal component analysis* (PCA)—one of the most popular dimensionality reduction methods—the axes of this new coordinate system are the axes of maximum variance of the training data. Alternatively, some nonlinear and more discriminative dimensionality reduction algorithms are specifically aimed at preserving the local information in the transformation, as in *locally linear embedding* [**78**] and *Laplacian eigenmaps* [**4**]. In all cases, samples are transformed into the new coordinate system and classification is performed in this new space.

  The idea behind dimensionality reduction is that the representation of the data set in the new space better reveals its structure. Further, since the dimension of the new space is smaller than that of the original space, analysis of the data (e.g., classification) becomes more computationally efficient.

- *Local feature extraction*: These methods are designed to pick out portions of the data samples that contain some critical information. Once again, classification is performed using the extracted features instead of the original images. In facial recognition, for example, the eyes, nose, and mouth portions of an image contain much of the person's identity, and so a local feature extractor might aim to classify the data set based solely on this information.

  To achieve good classification in general, the extracted local features must be robust to variants such as scaling, rotation, and illumination conditions (when the sample represents an image). The *scale-invariant feature transform* (SIFT) [**63**], for example, determines key locations from training images by extracting feature vectors from image "blobs" that

appear as each image is increasingly blurred. These feature vectors are designed to be invariant to various transformations. Other local feature extractors include *histograms of oriented gradients* [**24**] and *local binary patterns* [**70**].

- *Discriminant feature extraction*: Not necessarily disjoint from the above two categories, these types of feature extractors are designed specifically for the task at hand, e.g., compression or noise removal, or in our case, classification. For example, the *local discriminant basis selection algorithm* [**79**, **80**] can be used to select the basis (i.e., transformation matrix) from a dictionary of bases that minimizes the pairwise divergence between classes in the transform space, thus improving classification. If the dictionary of bases used is such that the expansion coefficients (i.e., the extracted features) represent local aspects of the original data samples (e.g., wavelet bases), then this method can be categorized as local feature extraction [**79**, **80**]. Further, if only the most discriminative basis elements are used, this method performs dimensionality reduction.

Though our focus in this dissertation is on classification algorithms and not feature extraction, we remark that the choice of feature extractor can be as important as that of the classifier. Simple methods such as kNN can achieve surprisingly good performance with the right pre-processing and feature extraction methods, and classification results can usually be greatly improved by putting thought and effort into choosing the best features. This should be kept in mind in Chapters 5 and 7, in which our experiments utilize only basic PCA for dimensionality reduction.

2.3. The Class Manifold Assumption

One approach to designing a classification algorithm begins by assuming that the data in each class lies on (or close to) a smooth, low-dimensional manifold. There is significant precedence for this assumption: In face recognition, for example, it has been argued by Chang et al. that, since people change expression continuously over time, this assumption holds on the set of facial images that vary in expression [**20**]. A similar argument was made by Seung and Lee based on the smoothness of facial rotation [**81**]. It has also been shown that the space of all images of a fixed subject under varying illumination conditions can be well-approximated by a low-dimensional linear subspace [**59**]; see Chapter 11 for further details. Additional discussion and references regarding the manifold structure

of face databases are contained in Chapters 4 and 5. In the case of handwritten digit classification, Simard et al. modeled the set of all variations of a sample as the output of a differentiable function [**85**] (see Section 3.1.4), and Yang et al. applied this class manifold assumption not only to the recognition of handwritten digits but also to vehicle classification [**110**]. There are other examples.

In the class manifold approach to classification, the training data is used to (either explicitly or implicitly) produce a model of each class manifold, and the test sample is then classified based on a method of determining best fit. Though some of the classification algorithms we discuss do not explicitly use this procedure, we assume throughout this dissertation that the data we wish to classify has this structure. Locally, these smooth, low-dimensional class manifolds look like Euclidean space, and it is sufficient for the reader to imagine, for example, the topology of a golf course, in order to understand our use of this term in subsequent chapters.

CHAPTER 3

Representation-Based Classification

This chapter reviews relevant methods of representation-based classification and discusses their theoretical motivations and empirical efficacy.

3.1. Per-Class Decomposition

In this section, we consider representation-based classifiers that compute an approximation or representation of the test sample using the training samples in each class separately.

3.1.1. 1-Nearest Neighbors. *k-nearest neighbors* (kNN) can be viewed as a representation-based classifier when Euclidean distance is used and $k = 1$. Recall that we classify $\boldsymbol{y}$ in 1NN by finding its nearest sample in the training set and then assigning $\boldsymbol{y}$ to the class of that training sample. This is equivalent to solving

$$\text{class\_label}(\boldsymbol{y}) = \arg \min_{1 \leq l \leq L} \left\{ \min_{1 \leq j \leq N_l} \left\| \boldsymbol{y} - \boldsymbol{x}_j^{(l)} \right\|_2 \right\}, \tag{3.1}$$

where the set $\{\boldsymbol{x}_1^{(l)}, \ldots, \boldsymbol{x}_{N_l}^{(l)}\}$ contains the N_l training samples in class l, $l = 1, \ldots, L$. From this minimization problem, we can see that 1NN works by determining the best approximation of $\boldsymbol{y}$ using a *single* training sample.

This method is extremely popular in practice due to its simplicity. Further, it has been shown that as the number of samples N_{tr} goes to infinity, the probability of 1NN computing the wrong classification assignment is bounded above by two times the Bayes probability of error (the lowest possible error rate based on the probability distribution of the samples) [**23**]. However, 1NN has critical limitations. Because it considers only a single training sample at a time, it does not utilize information contained in each class as a whole. This makes 1NN very sensitive to noise, outliers, and the number of training samples in each class; the class assignment can easily change given a small perturbation of the data samples or the addition of new training data. Secondly, as we

mentioned in Chapter 2, the classification process in 1NN tells us no more than the class assignment itself: it gives us no indication of the relationship between the test sample and its assigned class (except that it contains a training sample relatively near $\boldsymbol{y}$).

3.1.2. Nearest Subspace Classification. *Nearest subspace classification* (NSC) works under the assumption that the correct class of a test sample is the class whose training set can approximate it with the least ℓ^2-error. The distance between the test sample and its projection onto the span of the training samples in each class is computed, with classification based on minimal distance:

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\le l\le L}\Big\{\min_{\boldsymbol{\alpha}_l\in\mathbb{R}^{N_l}} \big\|\boldsymbol{y} - X^{(l)}\boldsymbol{\alpha}_l\big\|_2\Big\}. \tag{3.2}$$

Here, $X^{(l)} := [\boldsymbol{x}_1^{(l)}, \ldots, \boldsymbol{x}_{N_l}^{(l)}]$ is the matrix that contains the training samples from class l, $l = 1, \ldots, L$. In the case that the columns of $X^{(l)}$ are linearly independent (which requires that the sample dimension m is greater than the number of class l samples N_l), the solution to Eq. (3.2) is given by

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\le l\le L} \|\boldsymbol{y} - \hat{\boldsymbol{y}}_l\|_2^2, \tag{3.3}$$

where $\hat{\boldsymbol{y}}_l := X^{(l)}((X^{(l)})^{\mathsf{T}}X^{(l)})^{-1}(X^{(l)})^{\mathsf{T}}\boldsymbol{y}$.

As discussed by Zhang et al. [**114**], NSC is sensitive to small errors in the data samples. In face recognition, for example, if two subjects look fairly similar to each other, it may be the case that

$$\min_{\boldsymbol{\alpha}_l\in\mathbb{R}^{N_l}} \big\|\boldsymbol{y} - X^{(l)}\boldsymbol{\alpha}_l\big\|_2 \approx \min_{\boldsymbol{\alpha}_{l'}\in\mathbb{R}^{N_{l'}}} \big\|\boldsymbol{y} - X^{(l')}\boldsymbol{\alpha}_{l'}\big\|_2,$$

causing NSC to have trouble differentiating the lth and (l')th classes. Further, if the training class sizes are large enough ($N_l > m$), then there is an infinite number of solutions $\boldsymbol{\alpha}_l$ to Eq. (3.2), i.e., $\hat{\boldsymbol{y}}_l = \boldsymbol{y}$ in Eq. (3.3). In both cases, the performance of NSC can be improved by adding *regularization*.

3.1.3. Regularized Nearest Subspace. In *regularized nearest subspace* (RNS) [**115**], Eq. (3.2) in NSC is replaced with

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\le l\le L}\Big\{\min_{\boldsymbol{\alpha}_l\in\mathbb{R}^{N_l}} \big\|\boldsymbol{y} - X^{(l)}\boldsymbol{\alpha}_l\big\|_2^2 + \lambda\|\boldsymbol{\alpha}_l\|_p\Big\}, \tag{3.4}$$

where $p \in \{1, 2\}$ and $\lambda > 0$ is the trade-off between approximation and regularization. Here, the regularization term $\|\boldsymbol{\alpha}_l\|_p$ forces the coefficients in the approximation to be small.

As we discuss below, the test sample $\boldsymbol{y}$ may not be close to the span of the training samples in the correct class, causing the projection distances in NSC and RNS to be poor indicators of class. This is especially likely to occur if the training sets are undersampled. The following method is less sensitive to this weakness.

3.1.4. Tangent Distance Classification. The motivation behind *tangent distance-based classification* (TDC) [**18**,**85**,**110**] is as follows: Given a sample $\boldsymbol{x}_j \in \mathbb{R}^m$ and a vector of deformation parameters $\boldsymbol{\gamma}$,[1] we consider the transformed sample $\tilde{\boldsymbol{x}}_j = S(\boldsymbol{x}_j, \boldsymbol{\gamma})$ and the set of small deformations of $\boldsymbol{x}_j$ given by $S_{\boldsymbol{x}_j} := \{\tilde{\boldsymbol{x}}_j \,|\, \exists\, \boldsymbol{\gamma} \text{ for which } \tilde{\boldsymbol{x}}_j = S(\boldsymbol{x}_j, \boldsymbol{\gamma})\}$ [**85**]. In the case that the sample represents an image of an object, we can think of $S_{\boldsymbol{x}_j}$ as the space of various "versions" of the object, for example, the space containing all the different ways the digit "2" can be written by hand.

TDC associates each class with one or more spaces $S_{\boldsymbol{x}_j}$ and then aims to classify the test sample to the class associated with the closest such space. In the basic TDC algorithm, this is done by determining a linear approximation to each $S_{\boldsymbol{x}_j}$ and then measuring the distance between these tangent hyperplanes and the test sample, with classification to the class with the closest hyperplane. Simard et al. [**85**] construct these tangent hyperplanes from a set of predetermined transformations of selected prototypes for each class. This is a good approach when there is prior knowledge available regarding the deformation parameters, as in the case of the handwritten digits used by Simard et al. [**85**].

However, such prior information is generally not available. To address this issue, the local TDC algorithm of Yang et al. [**110**] determines the nearest neighbors of the given test sample in each class, scales them so that $\boldsymbol{y}$ approximates their mean, and then uses the scaled neighbors to construct a class-specific sample covariance matrix. The authors state that if the (scaled) neighbors are i.i.d. on a sufficiently-smooth manifold and the covariance matrix has full-rank, then the eigenvectors of the covariance matrix form a basis for the tangent hyperplane on the class manifold at the mean

[1]For example, consider $\boldsymbol{\gamma} \in \mathbb{R}^3$ with first coordinate representing rotation angle and second and third coordinates representing vertical and horizontal translation, respectively. These deformation parameters are applicable to tasks such as the classification of handwritten digits.

of the scaled neighbors. The Euclidean distance between this hyperplane and the test sample can then be measured.

TDC is a representation-based method in the sense that this hyperplane can be viewed as the affine span of *newly-generated* training samples, i.e., approximate tangent hyperplane basis vectors. We discuss this further in Chapter 4. Though TDC methods can be effective, the accuracy of tangent vectors computed from training data is very sensitive to the amount of noise in the data samples. Further, most methods for computing these tangent vectors require user-specified parameters. Thus cross-validation must often be performed to estimate them (increasing the algorithm's overall runtime), with improper setting of these parameters negatively affecting the algorithm's performance.

We formally state two implementations of TDC in Chapter 5.

3.1.5. Shortcomings of Per-Class Decomposition Methods. In the case of limited training data, per-class approximations of the test sample may be poor, even over the correct class. Thus no class may distinguish itself under this framework. Even with the addition of tangent vectors (as in TDC), class-specific training data may fail to provide a good estimation of a generic test sample. Though these per-class methods generally perform more competitively when $m < N_l$ and regularization is added (i.e., RNS), there are important applications in which we do not have the luxury of such large training sets. For example, face recognition is generally a small class size problem, with only a handful of training images available in each class. Further, the raw number of pixels in these images can be very large. Even with dimensionality reduction or feature extraction, it is challenging (and sometimes impossible) to satisfy the inequality $m < N_l$, $1 \leq l \leq L$, while still retaining enough information in the extracted features to distinguish the classes.

The following algorithm is an alternative to per-class decomposition methods.

3.2. Sparse Representation-Based Classification

Sparse representation-based classification (SRC) [**104**] plays a central role in this dissertation. This method decomposes the test sample over the entire training set, solving the optimization

problem

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\text{tr}}}} \|\boldsymbol{\alpha}\|_1, \text{ subject to } \boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}. \tag{3.5}$$

It is assumed that the training samples have been normalized to have ℓ^2-norm equal to 1, so that the representation in Eq. (3.5) will not be affected by the samples' magnitudes. The use of the ℓ^1-norm in the objective function is designed to approximate the ℓ^0-"norm," i.e., to aim at finding the smallest number of training samples that can accurately represent the test sample $\boldsymbol{y}$. It is argued that the nonzero coefficients in the representation will occur primarily at training samples in the same class as $\boldsymbol{y}$, so that

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\le l\le L} \left\|\boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}^*)\right\|_2 \tag{3.6}$$

produces the correct class assignment. Here, δ_l is the indicator function that acts as the identity on all coordinates corresponding to samples in class l and sets the remaining coordinates to zero. In other words, $\boldsymbol{y}$ is assigned to the class whose training samples contribute the most to the sparsest representation of $\boldsymbol{y}$ over the entire training set.

The reasoning behind this is the following: It is assumed that the class manifolds are linear subspaces, so that if each class's training set contains a spanning set of the corresponding subspace, the test sample can be expressed as a linear combination of training samples in its ground truth class. If the number of training samples in each class is small relative to the number of total training samples N_{tr}, this representation is naturally sparse [**104**].

As real-world data are often corrupted by noise, the constrained ℓ^1-minimization problem in Eq. (3.5) may be replaced with its regularized version

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\text{tr}}}} \left\{\frac{1}{2}\|\boldsymbol{y} - X_{\text{tr}}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1\right\}. \tag{3.7}$$

Similar to RNS, λ is the trade-off between error in the approximation and the sparsity of the coefficient vector. We summarize SRC in Algorithm 1.

There have been many modifications to the original SRC algorithm addressing various issues, such as its high computational complexity [**60**, **114**] and its requirement that training samples be normalized [**116**, **118**]. It has also been mentioned that for SRC to work correctly in theory,

Algorithm 1 Sparse Representation-Based Classification (SRC) [**104**]

Input: Matrix of training samples $X_{\text{tr}} \in \mathbb{R}^{m \times N_{\text{tr}}}$, test sample $\boldsymbol{y} \in \mathbb{R}^m$, number of classes L, and error/sparsity trade-off λ.

Output: The computed class label of $\boldsymbol{y}$: class_label($\boldsymbol{y}$).

1: Normalize each column of X_{tr} to have ℓ^2-norm equal to 1.
2: Use an ℓ^1-minimization algorithm to solve either the constrained problem in Eq. (3.5) or the regularized problem in Eq. (3.7).
3: **for** each class $l = 1, \ldots, L$, **do**
4: Compute the norm of the class l residual: $\text{err}_l(\boldsymbol{y}) := \left\|\boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}^*)\right\|_2$. Set class_label($\boldsymbol{y}$) $= \arg\min_{1 \le l \le L}\{\text{err}_l(\boldsymbol{y})\}$.
5: **end for**

the class manifolds must be disjoint so that no two classes may each emit a representation of the given test sample (see, e.g., the work of Hui et al. [**53**]). But in face recognition, for example, similar looking subjects (e.g., siblings) and extreme lighting conditions are two of the many reasons class manifolds may intersect. The objection to SRC is that, given that $\boldsymbol{y}$ is in the intersection of two linear subspaces, the number of nonzero coefficients may not be able to distinguish the difference between the two possible representations of $\boldsymbol{y}$, resulting in poor classification. Though this argument assumes that the constrained optimization problem in Eq. (3.5) is used, an analogous objection holds for the regularized version in Eq. (3.7) when the test sample is sufficiently close to more than one class manifold.

This argument is related to another critique of SRC that questions whether or not the concept of sparsity is necessary, or even helpful, in classification. The phrase "sparsity does not enforce locality" has been used to point out that significantly-contributing training samples in the sparse representation of $\boldsymbol{y}$ may be far away from $\boldsymbol{y}$ with respect to distance metrics such as Euclidean distance [**101**]. Many arguments claim that the ability to enforce locality in a representation (by restricting large coefficients to dictionary elements close to $\boldsymbol{y}$) is necessary for correct classification, and thus the sparseness prior in SRC alone is insufficient [**101**].

Though we discuss these issues, particularly the role of sparsity in SRC, more in Chapters 4, 5, 12, and 14, we make a couple of comments here:

(1) The effectiveness of ℓ^1-minimization in SRC may not be solely based on its ability to recover in many situations the sparsest solution (see Chapter 9 for conditions that guarantee this so-called ℓ^1/ℓ^0-equivalence). More broadly, its efficacy may lie in its ability to promote

sparsity *as well as locality* in the approximation/representation [**109**]. Thus a built-in criterion of closeness may break ties when multiple, class-specific sparse representations of the test sample exist. For example, when the constraint $\mathbf{1}^\mathsf{T}\boldsymbol{\alpha} = 1$ is added to the optimization problem in Eq. (3.5), Yang et al. [**109**] showed the following: If $\overline{\boldsymbol{x}}_{\boldsymbol{\alpha}}$ denotes the mean of the training samples with nonzero coefficients in the representation $\boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}$ (note that in general, these training samples will not all be in the same class), then ℓ^1-minimization finds the representation of $\boldsymbol{y}$ such that the ℓ^1-distance between $\boldsymbol{y}$ and $\overline{\boldsymbol{x}}_{\boldsymbol{\alpha}}$ is minimized, i.e., $\|\boldsymbol{y} - \overline{\boldsymbol{x}}_{\boldsymbol{\alpha}^*}\|_1$ is minimal. See Yang et al.'s paper [**109**] for details, as well as our findings in Chapter 12.

(2) Let us suppose that the assumptions in SRC hold, namely, that (i) the class manifolds are linear subspaces spanned by their training samples and that (ii) the number of samples in each training class is small, relative to the total number of training samples. Further, assume that (iii) these class subspaces do not intersect (except for at the origin). Then sparsity in the context of SRC is a legitimate method to find the representation of $\boldsymbol{y}$ in terms of the training samples in its ground truth class, and thus leads to correct classification. That is, assuming that the strict assumptions (i)-(iii) hold, no specific notion of locality is needed.

3.3. Similar Methods

3.3.1. Collaborative Representation. In addition to its sparsity component, SRC is fundamentally different from the per-class methods discussed in Section 3.1 in that it represents (or approximates) a test sample using all the classes at the same time. This in itself is a viable solution to the shortcomings of per-class decomposition methods [**114**]. In this global representation, the classes all "collaborate" in expressing $\boldsymbol{y}$, hence the term *collaborative representation*, as coined by Zhang et al. [**114**]. When $m < N_{\text{tr}}$ (which is much more practical than requiring $m < N_l$, especially when the number of classes is large), there is an infinite number of exact representations of $\boldsymbol{y}$ in terms of the set of training samples, and we can make the problem well-posed by adding a suitable regularization term or constraint on the coefficient vector. In SRC, this is ℓ^1-minimization.

Collaborative representation is based on the idea that samples in the data set, independent of their classes, have common properties. For example, consider the commonalities between facial images—each is roughly the same shape with eyes, a nose and mouth, and with the majority of pixels corresponding to areas such as skin and hair that are primarily devoid of significant class information. In a collaborative representation, this communal information can be shared over all the classes. In contrast, the discriminative information contained in the test image, unique to its ground truth class, must be contributed by training samples in this same class. Thus these samples (more specifically, their coefficients) stand out in the overall representation. It follows that a class assignment can be made by comparing the coefficients that correspond to samples in each class. Zhang et al. argue that this collaborative mechanism in SRC, and not necessarily its connection to sparsity, is fundamental to its success, at least on some data sets [**114**].

Though we extensively investigate the connection between SRC and sparsity throughout this dissertation, the following example specifically illustrates the efficacy of the collaborative representation aspect in SRC as compared to a per-class approach discussed in Section 3.1: We took the first $L = 20$ classes of AR-2, a version of the AR Face Database [**65**] discussed in Chapter 5. Images in AR-2 contain faces with differing expression and lighting conditions, as well as a small number of images of each subject wearing: (i) sunglasses, or (ii) a scarf covering the nose and mouth. These are considered natural occlusions. After randomly selecting one from the 20 classes, we chose a test sample with scarf occlusion from that class, and then purposely removed all images of that subject with scarf occlusion from the training set. The idea is to try to trick the classification algorithm: considering that the scarf occlusion is quite major, will the algorithm select an incorrect class that contains a scarf image, or will it be able to identify the actual subject based on the portion of the test image that is not covered by the scarf?

We compared RNS (with $p = 1$ in Eq. (3.4)) and SRC (with the optimization problem formulated as in Eq. (3.7)). The parameter λ was set to 0.001 in each method. Since both algorithms utilize the sparsity-promoting ℓ^1-norm, this allowed us to directly compare per-class and collaborative approaches to representation-based classification.

RNS was unable to correctly classify the test image, and instead selected an incorrect class with a scarf-occluded image and placed a large coefficient at this image, as illustrated in Figure

3.1. Because many of the classes contained scarf images in their training sets, the class residuals in RNS, given by

$$\min_{\boldsymbol{\alpha}_l \in \mathbb{R}^{N_l}} \left\{ \left\| \boldsymbol{y} - X^{(l)} \boldsymbol{\alpha}_l \right\|_2^2 + \lambda \|\boldsymbol{\alpha}_l\|_1 \right\}, \quad 1 \leq l \leq L, \tag{3.8}$$

were quite similar to each other. This indicates a low level of confidence in the classification decision, since RNS classifies $\boldsymbol{y}$ to the class with the minimum residual (see Eq. (3.4)). We illustrate this in Figure 3.3a.

In contrast, the collaborative representation mechanism in SRC was able to identify the correct class. The coefficient vector in SRC had nonzeros at a handful of scarf images over a number of classes (i.e., the classes *collaborated* to represent this portion of the test image), as well as nontrivial coefficients at multiple un-occluded images from the correct class. This is illustrated in Figure 3.2. Since the coefficients corresponding to scarf images were spread out among the classes, this latter type of coefficient—those at un-occluded images in the correct class—produced a very confident classification decision: the class with the smallest residual $\text{err}_l(\boldsymbol{y})$ (defined in Algorithm 1) was clearly differentiated from the remaining classes, as illustrated in Figure 3.3b.

(a) Test sample

(b) coeff = 0.88

FIGURE 3.1. An example of the per-class decomposition approach in RNS failing. RNS places a very large coefficient on a scarf image from the wrong class, resulting in incorrect classification. For reference, note that all other coefficients in this class approximation had magnitudes less than 0.1.

The basic model of Zhang et al.'s *collaborative representation-based classification* (CRC) [**114**], designed to generalize the multi-class representation approach in SRC, is the following: Assuming that the training samples have been normalized to have ℓ^2-norm equal to 1, the optimization

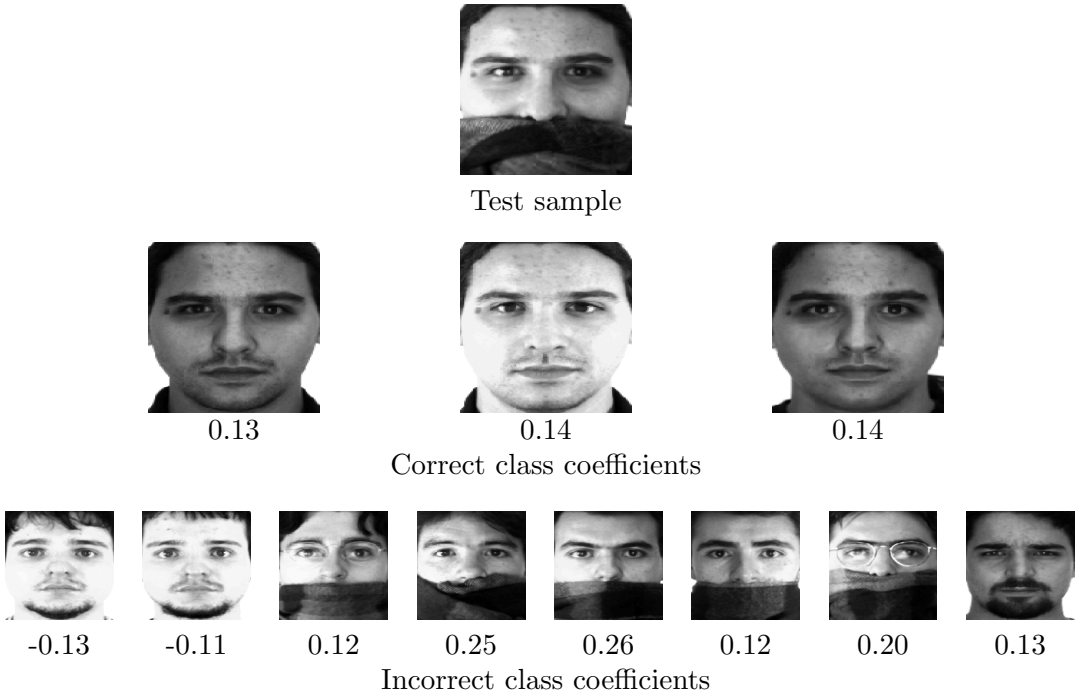


FIGURE 3.2. An example of the collaborative-representation mechanism in SRC. To decompose the occluded test sample (top row), training samples from incorrect classes collaborated to represent the scarf portion of the image (bottom row). The correct class was needed to provide class-specific details found in the un-occluded portion of the test image (middle row), resulting in correct classification. Training samples with coefficient magnitude less than 0.1 have been omitted.

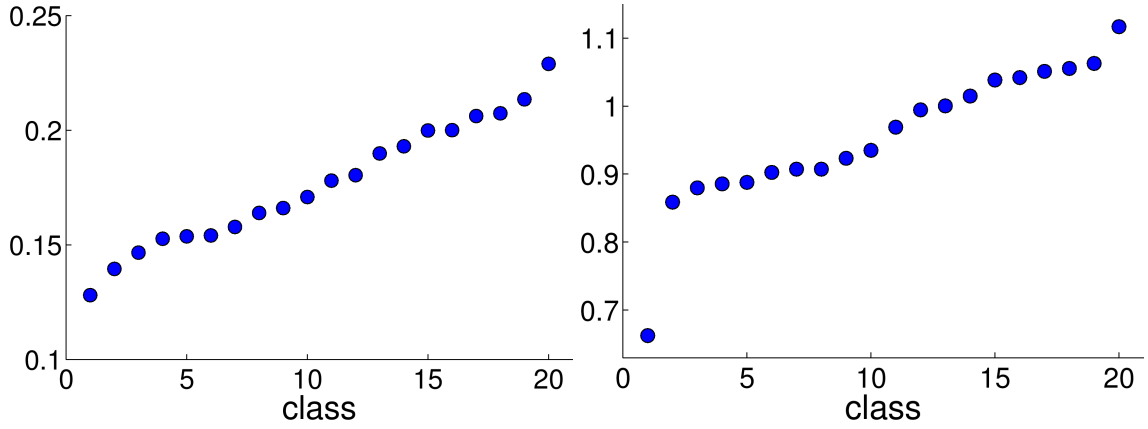
problem in CRC is

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\mathrm{tr}}}} \left\{ \|\boldsymbol{y} - X_{\mathrm{tr}}\boldsymbol{\alpha}\|_q^2 + \lambda\|\boldsymbol{\alpha}\|_p^2 \right\}, \tag{3.9}$$

with classification by

$$\text{class label}(\boldsymbol{y}) = \arg\min_{1\le l\le L} \left\{ \frac{\|\boldsymbol{y} - X_{\mathrm{tr}}\delta_l(\boldsymbol{\alpha}^*)\|_2}{\|\delta_l(\boldsymbol{\alpha}^*)\|_p} \right\}. \tag{3.10}$$

Here, $p \in \{1, 2\}$ and $q = 2$, unless the data is occluded/corrupted, in which case $q = 1$. As in SRC, δ_l is the indicator function that acts as the identity on all coordinates corresponding to samples in class l and sets the remaining coordinates to zero. The normalization factor $\|\delta_l(\boldsymbol{\alpha}^*)\|_p$ in CRC is said to contain discriminative class information in addition to that contained in the class residual $\|\boldsymbol{y} - X_{\mathrm{tr}}\delta_l(\boldsymbol{\alpha}^*)\|_2$ [**114**].



(a) RNS class residuals (sorted)

(b) SRC class residuals (sorted)

FIGURE 3.3. Comparison of the class residuals in RNS (from Eq. (3.8)) and SRC ($\text{err}_l(\boldsymbol{y})$, $1 \leq l \leq L$, from Algorithm 1) in classification of a scarf image from AR-2 when no same-class scarf images are included in the training set. (a) The class residuals in RNS are quite similar to each other, indicating low confidence in the (incorrect) classification decision. (b) SRC discriminates the correct class with confidence. Classes have been reordered for best visualization.

With the exception of the normalization factor in Eq. (3.10), SRC is essentially CRC with $p = 1$ and $q = 2$.[2] Since there is no closed-form solution to Eq. (3.9) for $p = 1$, it can be expensive to solve, especially for large m and N_{tr}.

Designed to address the high computational complexity of SRC, *collaborative representation-based classification with regularized least squares* (CRC-RLS) [**114**] is the case that $p = q = 2$. Though the use of the ℓ^2-norm does not enforce sparsity, it serves to prevent the coefficients from blowing up. Also, as in RNS (and SRC), this regularization makes the solution more stable to small errors in the data samples.

There is a closed-form solution for $\boldsymbol{\alpha}^*$ in CRC-RLS, given by

$$\boldsymbol{\alpha}^* = (X_{\text{tr}}^{\mathsf{T}} X_{\text{tr}} + \lambda I)^{-1} X_{\text{tr}}^{\mathsf{T}} \boldsymbol{y}.$$

In practical applications, this makes the algorithm extremely efficient. Instead of solving Eq. (3.9) for each test sample at a time, we can set $P := (X_{\text{tr}}^{\mathsf{T}} X_{\text{tr}} + \lambda I)^{-1} X_{\text{tr}}^{\mathsf{T}}$ in the offline stage of CRC-RLS. Then, to classify a given test sample, we need only compute $\boldsymbol{\alpha}^* = P\boldsymbol{y}$ and plug it into

[2]When $p = q = 1$, this is equivalent to what we call the *occlusion version* of SRC [**104**]. We describe this modification in Section 5.4.3.

Eq. (3.10). Further, the CRC authors show experimentally that CRC-RLS can achieve classification rates comparable to those in the $p = 1$ case, provided that the feature dimension m is large enough [**114**,**115**]. In fact, in the example discussed above, CRC-RLS (like SRC) correctly classified the AR-2 test sample with scarf occlusion, due to its collaborative representation approach. Note that CRC-RLS is not specifically aimed at increasing the classification accuracy of SRC; instead, its purpose is to illustrate that there are situations in which the computational complexity of SRC can be significantly decreased without a worsening in classification decisions.

3.3.2. Locality-Sensitive Dictionary Learning SRC. The objective of *locality-sensitive dictionary learning for SRC* (LSDL-SRC) [**101**] is to modify SRC in a way that helps ensure that large coefficients in the approximation of a given test sample will be placed at training samples that are close (in terms of either ℓ^2-distance or an exponential distance function) to the test sample, hence incorporating a locality constraint into the SRC algorithm.

In the dictionary learning phase of LSDL-SRC, a subset of the training samples are selected based on their ability to accurately and locally approximate a generic test sample (as implicitly approximated by the training data, so that the dictionary can be learned offline). Let $X^{(l)} \in \mathbb{R}^{m \times N_l}$ contain the class l training samples and $D^{(l)} \in \mathbb{R}^{m \times K_l}$ be the portion of the learned dictionary corresponding to class l. Here, $K_l \in \mathbb{N}$, $1 \leq l \leq L$, are user-specified parameters that determine the dictionary size $K := \sum_{l=1}^{L} K_l$. Set $\boldsymbol{\alpha}_j^{(l)} \in \mathbb{R}^{K_l}$ to contain the decomposition coefficients that result when the jth training sample in the lth class is decomposed over $D^{(l)}$, and store these coefficient vectors in the matrix $A^{(l)} := [\boldsymbol{\alpha}_1^{(l)}, \ldots, \boldsymbol{\alpha}_{N_l}^{(l)}] \in \mathbb{R}^{K_l \times N_l}$. Lastly, let $\boldsymbol{p}_j^{(l)} \in \mathbb{R}^{K_l}$ be the vector whose kth entry contains the distance between the jth training sample in the lth class and $\boldsymbol{d}_k^{(l)}$, the kth column of $D^{(l)}$. With "$\odot$" denoting coordinate-wise multiplication, the dictionary learning phase in LSDL-SRC [**101**] iteratively solves

$$\min_{D^{(l)}, A^{(l)}} \left\| X^{(l)} - D^{(l)} A^{(l)} \right\|_F^2 + \lambda_{\mathrm{DL}} \sum_{j=1}^{N_l} \left\| \boldsymbol{p}_j^{(l)} \odot \boldsymbol{\alpha}_j^{(l)} \right\|_2^2 \text{ subject to } \mathbf{1}^{\mathsf{T}} \boldsymbol{\alpha}_j^{(l)} = 1,\ 1 \leq j \leq N_l, \tag{3.11}$$

for each class $1 \leq l \leq L$.

To make this more transparent, consider that the kth coordinate of the jth term of the summation in Eq. (3.11) is

$$\text{dist}\left(\boldsymbol{x}_j^{(l)}, \boldsymbol{d}_k^{(l)}\right)\alpha_{jk}^{(l)}, \tag{3.12}$$

for a distance function $\text{dist}(\cdot,\cdot)$. The authors propose both Euclidean (or ℓ^2) distance

$$\text{dist}_{\ell^2}\left(\boldsymbol{x}_j^{(l)}, \boldsymbol{d}_k^{(l)}\right) := \left\|\boldsymbol{x}_j^{(l)} - \boldsymbol{d}_k^{(l)}\right\|_2$$

and an exponential distance function given by

$$\text{dist}_{\exp}\left(\boldsymbol{x}_j^{(l)}, \boldsymbol{d}_k^{(l)}\right) := \sqrt{\exp\left(\frac{\left\|\boldsymbol{x}_j^{(l)} - \boldsymbol{d}_k^{(l)}\right\|_2^2}{\sigma}\right)}. \tag{3.13}$$

The parameter σ must be specified. The second factor in Eq. (3.12), $\alpha_{jk}^{(l)}$, is the coefficient of $\boldsymbol{d}_k^{(l)}$ when $\boldsymbol{x}_j^{(l)}$ is decomposed over $D^{(l)}$. Thus if $\boldsymbol{x}_j^{(l)}$ and $\boldsymbol{d}_k^{(l)}$ are far apart, this coefficient is forced to be small, and conversely.

After solving Eq. (3.11) for each class $1 \leq l \leq L$, the dictionaries $D^{(l)}$ are concatenated to form one large dictionary $D \in \mathbb{R}^{m\times K}$. The general SRC/CRC framework can then be used to classify a test sample $\boldsymbol{y}$ using this new dictionary and replacing Eq. (3.7) with

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^K} \|\boldsymbol{y} - D\boldsymbol{\alpha}\|_2^2 + \lambda_{\text{SRC}}\|\boldsymbol{p}\odot\boldsymbol{\alpha}\|_2^2. \tag{3.14}$$

Here, $\boldsymbol{p}\in\mathbb{R}^K$ is the vector whose kth entry is the distance between $\boldsymbol{y}$ and $\boldsymbol{d}_k$, the kth column of D. Thus significant nonzero coefficients are encouraged to occur at dictionary elements close to $\boldsymbol{y}$. After finding the minimizing argument $\boldsymbol{\alpha}^*$ of Eq. (3.14), the final classification of $\boldsymbol{y}$ is carried out using Eq. (3.6), as in SRC.

The authors of LSDL-SRC argue that the sparsity-promoting ℓ^1-norm regularization term is no longer needed, as *locality implies sparsity* [**101**]. This allows for a closed-form solution for $\boldsymbol{\alpha}^*$ in Eq. (3.14) (aiding computational efficiency), though the dictionary learning phase still requires iterative optimization. In experiments, LSDL-SRC has been shown to achieve higher classification accuracy than SRC, often with a smaller (more discriminative) dictionary [**101**]. See Chapter 5 for our empirical comparison of the two methods.

3.4. Summary

We presented several representation-based classification algorithms and discussed their strengths and weaknesses. Per-class decomposition methods, such as NSC, RNS, and TDC, compute an approximation of the test sample using each of the L classes separately and then classify $\boldsymbol{y}$ based on the best approximation. Though these methods can be effective, all the approximations may be equally poor when $m > N_l$, $1 \leq l \leq L$. On the other hand, the case that $m < N_l$, $1 \leq l \leq L$, requires regularization (as in RNS), and this generous training set sampling is often an impractical assumption in applications such as face recognition.

Alternatively, collaborative representation-based methods, such as SRC, CRC-RLS, and LSDL-SRC, decompose the test sample over the entire training set at once. These algorithms only require the more feasible condition, i.e., that $m < N_{\text{tr}}$, and are designed to identify the correct class by locating training samples that contribute class-specific properties of the test sample to the representation. In SRC, ℓ^1-minimization is used to limit the number of nonzero coefficients in the representation, whereas CRC-RLS and LSDL-SRC prevent coefficients from blowing up through the use of the more computationally-efficient ℓ^2-norm. Further, LSDL-SRC is designed to restrict nonzero coefficients to training samples that are close to the given test sample, adding an aspect of locality (and some sparsity) to the global representation.

As we will see in Chapters 5 and 7, there are instances in which SRC's ℓ^1-regularization is much more effective in identifying training samples in the correct class than the methods in CRC-RLS and LSDL-SRC. However, even SRC may struggle when the training set is small and the class manifolds are nonlinear, so that the correct class provides a poor approximation of the test sample. This is the motivation behind the algorithm we propose in the next chapter.

CHAPTER 4

Local Principal Component Analysis SRC

In this chapter, we propose a classification algorithm that is an alternative to the methods discussed in Chapter 3, in particular, SRC.[1]

4.1. Motivation

SRC explicitly makes two substantial assumptions: (i) class manifolds are *linear subspaces*, and (ii) the training data in each class *spans* its corresponding linear subspace. These assumptions are often violated. For example, facial images that vary in pose and expression lie on nonlinear class manifolds [**50**, **78**], and as we have discussed, small training set size is one of the primary challenges in face recognition and classification as a whole.

However, SRC can still achieve high classification accuracy on data sets that violate the first of these assumptions, as demonstrated by Wright et al. [**104**] in experiments on the AR face database [**65**]. AR contains expression and occlusion variations that suggest the underlying class manifolds are nonlinear, yet SRC often outperformed SVM (support vector machines) on AR for a wide variety of feature extraction methods and feature dimensions [**104**]. As we have discussed, since SRC decomposes the test sample over the entire training set, components of the test sample not within the span of its ground truth class's training samples may be absorbed by training samples from other classes. A similar fail-safe occurs when the class manifolds (linear or otherwise) are sparsely-sampled, e.g., if the second assumption is violated. However, this reliance on "off-class" training samples to partially represent or approximate the test sample can lead to misclassification, especially when the class manifolds are close together (recall the discussion in Section 3.2 regarding intersecting manifolds). In the case where class manifolds are nonlinear and/or sparsely-sampled, so that it is impossible to approximate $\boldsymbol{y}$ well using only the training samples in its ground truth class, this approximation could conceivably be improved if we were able to increase the sampling density

[1] A paper on this algorithm has been submitted for publication and is available at arXiv.org [**100**].

around $\boldsymbol{y}$, "fleshing out" its local neighborhood on the (correct) class manifold. Our proposed classification algorithm aims to do just this.

4.2. Algorithm Statement

Our proposed algorithm, *local principal component analysis sparse representation-based classification* (LPCA-SRC), is essentially SRC with a modified dictionary. This dictionary is constructed in two steps: In the offline phase of the algorithm, we generate new training samples as a means of increasing the sampling density. Instead of the linear subspace assumption in SRC, we assume that class manifolds are well-approximated by local tangent hyperplanes. To generate new training samples, we approximate these tangent hyperplanes at individual training samples using *local principal component analysis* (local PCA), and then add (randomly-scaled and shifted versions of) the basis vectors of these tangent hyperplanes to the original training set. Naturally, the shifted and scaled tangent hyperplane basis vectors (herein referred to as "tangent vectors") inherit the labels of their corresponding training samples. The result is an amended dictionary over which a generic test sample can ideally be decomposed using samples that approximate a local patch on the correct class manifold. In the case that the class manifolds are sparsely-sampled and/or nonlinear, this allows for a more accurate approximation of $\boldsymbol{y}$ using training samples (and their computed tangent vectors) from the test sample's ground truth class. Even in the case that class manifolds are linear subspaces, this technique ideally increases the sampling density around $\boldsymbol{y}$ on its (unknown) class manifold so that it may be expressed in terms of *nearby* samples.

In the online phase of LPCA-SRC, this extended training set is "pruned" relative to the given test sample, increasing computational efficiency and the locality of the resulting dictionary. Training samples (along with their tangent vectors) are eliminated from the dictionary if their Euclidean distances to the given test sample are greater than a threshold, and then classification proceeds as in SRC as the test sample is sparsely decomposed (via ℓ^1-minimization) over this local dictionary.

The method in LPCA-SRC has an additional benefit: When SRC is applied to the classification of high-resolution images (e.g., $> O(10^4)$ pixels), some method of dimensionality reduction is generally necessary to reduce the dimension of the raw samples, due to the high computational complexity of solving the ℓ^1-minimization problem. Basic dimensionality reduction methods such

as PCA (see Section 2.2) may result in the loss of class-discriminating details when the feature dimension is small. In Section 5.4.6, we show that the tangent vectors computed in LPCA-SRC can contain details of the raw images that have been lost in the PCA dimensionality reduction process.

The idea of using tangent hyperplanes for pattern recognition is not new. The most relevant instance of this is tangent distance classification (TDC) [**18**,**85**,**110**], as discussed in Chapter 3. On the other hand, there has been progress in addressing the limiting linear subspace assumption in SRC. For example, Ho et al. extended sparse coding and dictionary learning to general Riemannian manifolds [**52**]. Several "local" modifications of SRC implicitly ameliorate this issue; in *collaborative neighbor representation-based classification* [**99**] and previously-discussed LSDL-SRC, for instance, coefficients of the representation are constrained by their corresponding training samples' distances to the test sample, and so these algorithms need only assume linearity at the local level. Other modifications have aimed at enlarging the training set in SRC specifically for face recognition, for example, using virtual images that exploit the symmetry of the human face as in both the method of Xu et al. [**106**] and *sample pair based sparse representation classification* [**113**].

Our proposed algorithm does not require explicit knowledge of the class manifolds, as in Ho et al.'s work [**52**], and it is applicable to classification problems outside of face recognition. Further, it is designed to improve not only the locality but also the accuracy of the approximation of the test sample in terms of its ground truth class, and its sparse representation framework gives it an empirical advantage over methods based solely on tangent distance.

We formally state the offline and online portions of LPCA-SRC in Algorithms 2 and 3, respectively. Obviously, by the definition of "offline phase," the tangent vectors need only be computed once for any number of test samples.

4.3. Local Principal Component Analysis

In LPCA-SRC (in particular, Step 5 of Algorithm 2), we use the local PCA technique of Singer and Wu [**87**] to compute the tangent hyperplane basis $U^{(l,i)}$. We outline our implementation of their method in Algorithm 4. It computes a basis for the tangent hyperplane $T_{\boldsymbol{x}_i}\mathscr{M}$ at a point

Algorithm 2 Local PCA Sparse Representation-Based Classification (LPCA-SRC): **OFFLINE PHASE**

Input: $X_{\text{tr}} = [\boldsymbol{x}_1 \dots, \boldsymbol{x}_{N_{\text{tr}}}] \in \mathbb{R}^{m \times N_{\text{tr}}}$; number of classes L; local PCA parameters d (estimate of class manifold dimension) and n (number of neighbors).

Output: The normalized extended dictionary $D \in \mathbb{R}^{m \times (N_{\text{tr}}(d+1))}$; pruning parameter r.

Normalize the columns of X_{tr} to have ℓ^2-norm equal to 1.
for each class $l = 1, \dots, L$ **do**
  Let $\mathcal{X}^{(l)}$ be the set of class l training samples contained in X_{tr}.
  for each class l training sample $\boldsymbol{x}_i^{(l)}$, $i = 1, \dots, N_l$ **do**
    Use local PCA in Algorithm 4 with set of samples $\mathcal{X}^{(l)}$, selected sample $\boldsymbol{x}_i^{(l)}$, and parameters d and n to compute a basis $U^{(l,i)} := [\boldsymbol{u}_1^{(l,i)}, \dots, \boldsymbol{u}_d^{(l,i)}]$ of the tangent hyperplane approximation to the lth class manifold at $\boldsymbol{x}_i^{(l)}$. Store $U^{(l,i)}$ and $r_i^{(l)} := \|\boldsymbol{x}_{i_{n+1}}^{(l)} - \boldsymbol{x}_i^{(l)}\|_2$, the distance between $\boldsymbol{x}_i^{(l)}$ and its $(n+1)$st nearest neighbor in $\mathcal{X}^{(l)} \backslash \boldsymbol{x}_i^{(l)}$.
  end for
end for
Define the pruning parameter $r := \text{median}\{r_i^{(l)} \,|\, 1 \le i \le N_l,\, 1 \le l \le L\}$.
Initialize the extended dictionary $D = \emptyset$.
for each class $l = 1, \dots, L$ **do**
  for each class l training sample $\boldsymbol{x}_i^{(l)}$, $i = 1, \dots, N_l$ **do**
    Set $c := r\gamma$, $\gamma \sim \text{unif}(0,1)$, and form $\tilde{X}^{(l,i)} := \left[c\boldsymbol{u}_1^{(l,i)} + \boldsymbol{x}_i^{(l)}, \dots, c\boldsymbol{u}_d^{(l,i)} + \boldsymbol{x}_i^{(l)}, \boldsymbol{x}_i^{(l)}\right] \in \mathbb{R}^{m \times (d+1)}$.
    Normalize the columns of $\tilde{X}^{(l,i)}$ to have ℓ^2-norm equal to 1 and add it to the extended dictionary: $D = [D, \tilde{X}^{(l,i)}]$.
  end for
end for

$\boldsymbol{x}_i$ on the manifold $\mathscr{M}$, where it is assumed that the local neighborhood of $\boldsymbol{x}_i$ on $\mathscr{M}$ can be well-approximated by a tangent hyperplane of some dimension $d < m$. A particular strength of Singer and Wu's method is the weighting of neighbors by their Euclidean distances to the point $\boldsymbol{x}_i$, so that closer neighbors play a more important role in the construction of the local tangent hyperplane.

4.4. Remarks on the Choice of Parameters

In this section, we detail the roles of the parameters in LPCA-SRC and suggest strategies for estimating their optimal values.

4.4.1. Estimate of Class Manifold Dimension and Number of Neighbors. Recall that d is the estimated dimension of each class manifold and n is the number of neighbors used in local PCA. The number of samples in the smallest training class, denoted $N_{l_{\min}}$, limits the range of

Algorithm 3 Local PCA Sparse Representation-Based Classification (LPCA-SRC): **ONLINE PHASE**

Input: Test sample $\boldsymbol{y} \in \mathbb{R}^m$; normalized extended dictionary D; pruning parameter r; estimate of class manifold dimension d.

Output: The computed class label of $\boldsymbol{y}$: class_label($\boldsymbol{y}$).

1: Normalize $\boldsymbol{y}$ to have $\|\boldsymbol{y}\|_2 = 1$.
2: Initialize the pruned dictionary $D_{\boldsymbol{y}} = \emptyset$ and set $N_{\boldsymbol{y}} = 0$ (# of columns of $D_{\boldsymbol{y}}$).
3: **for** each class $l = 1, \ldots, L$ **do**
4: **for** each class l training sample $\boldsymbol{x}_i^{(l)}$, $i = 1, \ldots, N_l$ **do**
5: **if** $\left\|\boldsymbol{y} - \boldsymbol{x}_i^{(l)}\right\|_2 \leq r$ or $\left\|\boldsymbol{y} - (-\boldsymbol{x}_i^{(l)})\right\|_2 \leq r$ **then**
6: Add the portion $\tilde{X}^{(l,i)}$ of D corresponding to $\boldsymbol{x}_i^{(l)}$ and its tangent vectors to the pruned dictionary: $D_{\boldsymbol{y}} = [D_{\boldsymbol{y}}, \tilde{X}^{(l,i)}]$. Assign the columns of $\tilde{X}^{(l,i)}$ class l labels. Update $N_{\boldsymbol{y}} = N_{\boldsymbol{y}} + (d+1)$.
7: **end if**
8: **end for**
9: **end for**
10: Use an ℓ^1-minimization algorithm to compute the solution to the constrained problem

$$\boldsymbol{\alpha}^* := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\boldsymbol{y}}}} \left\{ \|\boldsymbol{\alpha}\|_1 \text{ s.t. } \boldsymbol{y} = D_{\boldsymbol{y}} \boldsymbol{\alpha} \right\} \tag{4.1}$$

or the regularized problem

$$\boldsymbol{\alpha}^* := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\boldsymbol{y}}}} \left\{ \frac{1}{2} \|\boldsymbol{y} - D_{\boldsymbol{y}} \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1 \right\}. \tag{4.2}$$

11: **for** each class $l = 1, \ldots, L$, **do**
12: Compute the norm of the class l residual: $\text{err}_l(\boldsymbol{y}) := \left\|\boldsymbol{y} - D_{\boldsymbol{y}} \delta_l(\boldsymbol{\alpha}^*)\right\|_2$. Set class_label($\boldsymbol{y}$) = $\arg\min_{1 \leq l \leq L} \{\text{err}_l(\boldsymbol{y})\}$.
13: **end for**

Algorithm 4 Local Principal Component Analysis (Local PCA, adapted from Singer and Wu [**87**])

Input: Set of samples $\mathcal{X}$, selected sample $\boldsymbol{x}_i \in \mathcal{X}$, dimension of tangent hyperplane d, number of neighbors n.

Output: The basis $U^{(l,i)}$ of the approximated tangent hyperplane at the point $\boldsymbol{x}_i$.

1: Find the $n+1$ nearest neighbors (with respect to Euclidean distance) of $\boldsymbol{x}_i$ in $\mathcal{X} \backslash \boldsymbol{x}_i$. Store the n nearest neighbors as columns of the matrix $X_i := [\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_n}]$ and use the $(n+1)$st nearest neighbor to define $\epsilon_{\text{PCA}} := \|\boldsymbol{x}_{i_{n+1}} - \boldsymbol{x}_i\|_2^2$.
2: Form the matrix $\overline{X}_i$ by centering the columns of X_i around $\boldsymbol{x}_i$: $\overline{X}_i := [\boldsymbol{x}_{i_1} - \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{i_n} - \boldsymbol{x}_i]$.
3: Form a diagonal weight matrix D_i based on the distance between each neighbor and $\boldsymbol{x}_i$ as follows: Let $D_i(j,j) = \sqrt{K\left(\frac{\|\boldsymbol{x}_{i_j} - \boldsymbol{x}_i\|_2}{\sqrt{\epsilon_{\text{PCA}}}}\right)}$, $j = 1, \ldots, n$, where K is the Epanechnikov kernel given by $K(u) := (1 - u^2)\boldsymbol{\chi}_{[0,1]}$.
4: Form the weighted matrix $B_i := \overline{X}_i D_i$.
5: Find the first d left singular vectors of B_i using singular value decomposition. Denote these vectors by $\boldsymbol{u}_1^{(i)}, \ldots, \boldsymbol{u}_d^{(i)}$. Set $U^{(l,i)} := [\boldsymbol{u}_1^{(i)}, \ldots, \boldsymbol{u}_d^{(i)}]$.

values for d and n that may be used. Specifically,

$$d \leq n < N_{l_{\min}} - 1. \tag{4.3}$$

This follows from the fact that each training sample must have at least $n + 1$ neighbors in its own class, with the dimension d of the tangent hyperplane being bounded above by the number of columns n in the weighted matrix of neighbors B_i.

While we suggest that n be set using cross-validation, there are many methods for determining d. One may use the multiscale SVD algorithm of Little et al. [**62**] or Ceruti et al.'s DANCo (*Dimensionality from Angle and Norm Concentration* [**16**]). Alternatively, d may be set using cross-validation, as we do in our experiments in Chapter 5. Empirically, we find that cross-validation often selects d smaller than the (expected) true class manifold dimension, and in these cases, increasing d from the selected value (i.e., increasing the number of tangent vectors used) does not significantly increase classification accuracy. We expect that the addition of even a small number of tangent vectors (those indicating the directions of maximum variance on their local manifolds, per the local PCA algorithm) is enough to improve the approximation of the test sample in terms of its ground truth class. Additional tangent vectors are often unneeded. Since the value of d largely affects LPCA-SRC's computational complexity, these observations suggest that when the true manifold dimension is large, it is better to underestimate it than overestimate it.

REMARK 4.4.1. *Certainly, the parameters d and n could vary per class, i.e., d and n could be replaced with d_l and n_l, respectively, for $l = 1, \ldots, L$. In face recognition, however, if each subject is photographed under similar conditions, e.g., the same set of lighting configurations, then we expect that the class manifold dimension is approximately the same for each subject. Further, without some prior knowledge of the class manifold structure, using distinct d and n for each class may unnecessarily complicate the setting of parameters in LPCA-SRC.*

4.4.2. Pruning Parameter. Recall that we only include a training sample $\boldsymbol{x}_i^{(l)}$ and its tangent vectors in the pruned dictionary $D_{\boldsymbol{y}}$ if $\boldsymbol{x}_i^{(l)}$ (or its negative) is in the closed Euclidean ball $\overline{B_m(\boldsymbol{y}, r)} \subset \mathbb{R}^m$ with center $\boldsymbol{y}$ and radius r. Thus r is a parameter that prunes the extended dictionary D to obtain $D_{\boldsymbol{y}}$. A smaller dictionary is good in terms of computational complexity, as the

ℓ^1-minimization algorithm will run faster. Further, we can obtain this computational speedup without (theoretically) degrading classification accuracy: If $\pm\boldsymbol{x}_i^{(l)}$ is far from $\boldsymbol{y}$ in terms of Euclidean distance, then it is assumed that $\pm\boldsymbol{x}_i^{(l)}$ is not close to $\boldsymbol{y}$ in terms of distance along the class manifold. Thus $\boldsymbol{x}_i^{(l)}$ and its tangent vectors should not be needed in the ℓ^1-minimized approximation of $\boldsymbol{y}$.

A deeper notion of the parameter r is to view it as a rough estimate of the local neighborhood radius of the data set. More precisely, r estimates the distance from a sample within which its class manifold can be well-approximated by a tangent hyperplane (at that sample). Given X_{tr} and n, r is automatically computed, as described in Algorithm 2. That is, we set r to be the median distance between each training sample and its $(n+1)$st nearest neighbor (in the same class), where n, the number of neighbors in local PCA, is used to implicitly define the local neighborhood. It follows that r is a robust estimate of the local neighborhood radius, as learned from the training data.

This also explains our choice for the tangent vector scaling factor $c = r\gamma$, where $\gamma \sim \text{unif}(0, 1)$. Multiplying each tangent hyperplane basis vector $\boldsymbol{u}_j^{(l,i)}$, $1 \leq j \leq d$, by this constant and then shifting it by its corresponding training sample $\boldsymbol{x}_i^{(l)}$ helps to ensure that the resulting tangent vector, included in the dictionary $D_{\boldsymbol{y}}$ if $\pm\boldsymbol{x}_i^{(l)}$ is sufficiently close to $\boldsymbol{y}$, lies in the local neighborhood of $\boldsymbol{x}_i^{(l)}$ on the lth class manifold.

REMARK 4.4.2. *If the test sample $\boldsymbol{y}$ is far from the training data, defining r as in Algorithm 2 may produce $D_{\boldsymbol{y}} = \emptyset$, i.e., there may be no training samples within that distance of $\boldsymbol{y}$. Thus to prevent this degenerate case, we use a slightly modified technique for setting r in practice. After assigning the median neighborhood radius $r_1 := \text{median}\{r_i^{(l)} \,|\, 1 \leq i \leq N_l,\, 1 \leq l \leq L\}$, we define r_2 to be the distance between the test sample $\boldsymbol{y}$ and the closest training sample (up to sign). We then define the pruning parameter $r := \max\{r_1, r_2\}$. In the (degenerate) case that $r = r_2$, the dictionary consists of the closest training sample and its tangent vectors, leading to nearest neighbor classification instead of an algorithm error. However, experimental results indicate that the pruning parameter r is almost always equal to the median neighborhood radius r_1, and so we leave this technicality out of the official algorithm statement to make it easier to interpret.*

4.4.3. Using Cross-Validation to Set Multiple Parameters. On data sets of which we have little prior knowledge, it may be necessary to use cross-validation to set multiple parameters in LPCA-SRC. Since grid search (searching through all parameter combinations in a brute-force manner) is typically expensive, we suggest that cross-validation be applied to the parameters n, λ, and d, consecutively in that order as needed. During this process, we recommend holding λ equal to a small, positive value (e.g., 0.001) and setting $d = 1$ until their respective values are determined.

Our reasoning for suggesting this procedure is the following: During experiments, we found that the LPCA-SRC algorithm can be quite sensitive to the setting of n, especially when there are many samples in each training class (since there are many possible values for n). This is expected, as the setting of n affects both the accuracy of the tangent vectors and the pruning parameter r. In contrast, LPCA-SRC is empirically fairly robust to the values of λ and d used, and further, setting $d = 1$ can result in quite good performance in LPCA-SRC, even when the true dimension of the class manifolds is expected to be larger.

4.5. Computational Complexity and Storage Requirements

4.5.1. Computational Complexity of SRC. When the ℓ^1-minimization algorithm HOMOTOPY [**33**] is used, it is easy to see that the computational complexity of SRC is dominated by this step. This complexity is $O(N_{\text{tr}}m\kappa + m^2\kappa)$, where κ is the number of HOMOTOPY iterations [**107**]. HOMOTOPY has been shown to be relatively fast and good for use in robust face recognition [**107**]. In our experiments in Chapter 5, we use it in all classification methods requiring ℓ^1-minimization.

4.5.2. Computational Complexity of LPCA-SRC. The computational complexity of the offline phase in LPCA-SRC (Algorithm 2) is

$$O\Big(m\sum_{l=1}^{L} N_l^2 + N_{\text{tr}}mn\Big), \tag{4.4}$$

whereas that of the online phase (Algorithm 3) is

$$O\Big(N_{\text{tr}}m + \frac{N_{\boldsymbol{y}}}{d}\log\Big(\frac{N_{\boldsymbol{y}}}{d}\Big) + N_{\boldsymbol{y}}m\kappa + m^2\kappa\Big). \tag{4.5}$$

Recall that $N_{\boldsymbol{y}}$ denotes the number of columns in the pruned dictionary $D_{\boldsymbol{y}}$. We note that the offline cost in Eq. (4.4) is based on the linear nearest neighbor search algorithm for simplicity; in practice there are faster methods. In our experiments, we used ATRIA (*Advanced Triangle Inequality Algorithm* [**67**]) via the MATLAB TSTOOL functions `nn_prepare` and `nn_search` [**68**]. The first function prepares the set of class l training samples $\mathcal{X}^{(l)}$ for nearest neighbor search at the onset, with the intention that subsequent runs of `nn_search` on this set are faster than simply doing a search without the preparation function. Other fast nearest neighbor search algorithms are available, for example, *k-d tree* [**5**]. The cost complexity estimates of these fast nearest neighbor search algorithms are somewhat complicated, and so we do not use them in Eq. (4.4). Hence, Eq. (4.4) could be viewed as the worst-case scenario.

Offline and online phases combined, the very worst-case computational complexity of LPCA-SRC is $O(N_{\text{tr}}^4)$, which occurs when the second-to-last term in Eq. (4.5) dominates: i.e., when (i) $N_{\boldsymbol{y}} \approx (d+1)N_{\text{tr}}$ (no pruning); (ii) $m \approx N_{\text{tr}}$ (large relative sample dimension); (iii) very large class manifold dimension estimate d, so that d is relatively close to N_{tr} (note that this requires very large N_l for $1 \leq l \leq L$ by Eq. (4.3), which implies that L has to be very small); and (iv) $\kappa \approx m$ (many HOMOTOPY iterations). For small κ and N_l, $1 \leq l \leq L$, and when the pruning parameter r results in small $N_{\boldsymbol{y}}$ relative to N_{tr}, then the computational complexity reduces to approximately $O(N_{\text{tr}}m)$.

4.5.3. Storage Requirements. The primary difference between the storage requirements for LPCA-SRC and SRC is that the offline phase of LPCA-SRC requires storing the matrix $D \in \mathbb{R}^{m\times(d+1)N_{\text{tr}}}$, which has a factor of $d+1$ as many columns as the matrix of training samples $X_{\text{tr}} \in \mathbb{R}^{m\times N_{\text{tr}}}$ stored in SRC. Hence the storage requirements of LPCA-SRC are $(d+1)$ times the amount of storage required by SRC.

CHAPTER 5

Experiments with LPCA-SRC

In this chapter, we test our proposed classification algorithm, LPCA-SRC, on one synthetic database and three popular face databases. For all data sets, we used HOMOTOPY to solve the regularized versions of the ℓ^1-minimization problems, i.e., Eq. (3.7) for SRC and Eq. (4.2) for LPCA-SRC, using version 2.0 of the L1 Homotopy toolbox [**1**].

5.1. Algorithms Compared

We compared LPCA-SRC to the original SRC, $\text{SRC}_{\text{pruned}}$ (a modification of SRC which we explain shortly), two versions of TDC (our implementations are inspired by Yang et al. [**110**]), LSDL-SRC [**101**], kNN, and *kNN over extended dictionary*.

- *SRC*$_{\text{pruned}}$: To test the efficacy of the tangent vectors in the LPCA-SRC dictionary, this modification of SRC prunes the dictionary of original training samples using the pruning parameter r, as in LPCA-SRC. $\text{SRC}_{\text{pruned}}$ is exactly LPCA-SRC without the addition of tangent vectors.
- *Tangent distance classification* (TDC1 and TDC2): We compared LPCA-SRC to two versions of TDC to test the importance of our algorithm's sparse representation framework. Both of our implementations begin by first finding a pruned matrix $D_{\boldsymbol{y}}^{\text{TDC}}$ that is very similar to the dictionary $D_{\boldsymbol{y}}$ in LPCA-SRC. In particular, $D_{\boldsymbol{y}}^{\text{TDC}}$ can be found using Algorithm 2 (omitting Step 1) and Steps 2-9 in Algorithm 3. Neither the training nor test samples are ℓ^2-normalized in the TDC methods; compared to the SRC algorithms, TDC1 and TDC2 are not sensitive to the energy of the samples. We emphasize that the resulting matrix $D_{\boldsymbol{y}}^{\text{TDC}}$ contains training samples that are nearby $\boldsymbol{y}$, as well as their corresponding tangent vectors.

  In TDC1, we then divide $D_{\boldsymbol{y}}^{\text{TDC}}$ into the "subdictionaries" $D_{\boldsymbol{y}}^{(l)}$, where $D_{\boldsymbol{y}}^{(l)}$ contains the portion of $D_{\boldsymbol{y}}^{\text{TDC}}$ corresponding to class l. The test sample $\boldsymbol{y}$ is next projected onto the

space spanned by the columns of $D_{\boldsymbol{y}}^{(l)}$ to produce the vector $\hat{\boldsymbol{y}}^{(l)}$, and the final classification is performed using

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\leq l\leq L} \left\|\boldsymbol{y} - \hat{\boldsymbol{y}}^{(l)}\right\|_2.$$

Our second implementation, TDC2, is similar. Instead of dividing $D_{\boldsymbol{y}}^{\text{TDC}}$ according to class, however, we split it up according to training sample, obtaining the subdictionaries $D_{\boldsymbol{y}}^{(l,i)}$, where $D_{\boldsymbol{y}}^{(l,i)}$ contains the original training sample $\boldsymbol{x}_i^{(l)}$ and its tangent vectors. It follows that each subdictionary in TDC2 has $d+1$ columns. The given test sample $\boldsymbol{y}$ is next projected onto the space spanned by the columns of $D_{\boldsymbol{y}}^{(l,i)}$ to produce $\hat{\boldsymbol{y}}_i^{(l)}$, a vector on the (approximate) tangent hyperplane at $\boldsymbol{x}_i^{(l)}$. The final classification is performed using

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\leq l\leq L} \left\{ \min_{1\leq i\leq N_l} \left\|\boldsymbol{y} - \hat{\boldsymbol{y}}_i^{(l)}\right\|_2 \right\}.$$

- *Locality-sensitive dictionary learning for SRC* (LSDL-SRC): Recall from Chapter 3 that LSDL-SRC replaces the regularization term in Eq. (3.7) of SRC with a term that forces large coefficients to occur only at dictionary elements that are close to the given test sample. We use the exponential distance function given in Eq. (3.13), as this has been shown to produce better empirical results than the ℓ^2-distance function [**101**].
- *k-nearest neighbors classification* (kNN): The test sample is classified to the most-represented class from among the nearest (in terms of Euclidean distance) k training samples (k is odd).
- *k-nearest neighbors classification over extended dictionary* (kNN-Ext): This is kNN over the columns of the (full) extended dictionary that includes the original training samples and their tangent vectors. Samples are not normalized at any stage.

5.2. Setting of Parameters

For the synthetic database, we used cross-validation at each instantiation of the training set to choose the best parameters n, λ, and d in LPCA-SRC. (Though the true class manifold dimension is known on this database, we cannot always assume that this is the case.) We optimized the parameters consecutively as described in Section 4.4.3, each over its own set of discrete values. The

values for n and d had to satisfy the inequalities in Eq. (4.3). We used the same approach for the parameter λ in SRC, the parameters n and λ in $\text{SRC}_{\text{pruned}}$, and the parameters n and d in the TDC algorithms. Finally, we used a similar procedure for the multiple parameters in LSDL-SRC (including its number of dictionary elements), and we also set k in kNN and kNN-Ext using cross-validation.

Our approach for the face databases was very similar, though in order to save computational costs, we set some parameter values according to previously published works. In particular, we set $\lambda = 0.001$ in LPCA-SRC, SRC, and $\text{SRC}_{\text{pruned}}$, as was used in SRC by Waqas et al. [**99**]. Additionally, we set most of the parameters in LSDL-SRC to the values used by its authors [**101**] on the same face databases, though we again used cross-validation to determine its number of dictionary elements.

5.3. Synthetic Database

5.3.1. Database Description. The following synthetic database is easily visualized, and its class manifolds are nonlinear (though well-approximated by local tangent planes) with many intersections. Thus it is ideal for empirically comparing LPCA-SRC and SRC. The class manifolds are sinusoidal waves normalized to lie on S^2, with underlying equations given by

$$x(t) = \cos(t + \phi),$$

$$y(t) = \sin(t + \phi),$$

$$z(t) = A \sin(\omega t).$$

We set $\omega = 3$ and $A = 0.5$, and we varied ϕ to obtain L classes. In particular, we set $\phi = 2\pi/(3l)$ for data in class $1 \leq l \leq L = 4$. For each training and test set, we generated the same number $N_0 = N_l$, $l = 1, \ldots, L$, of samples in each class by (i) regularly sampling $t \in [\,0, 2\pi)$ to obtain the points $\boldsymbol{p}(t) = [x(t), y(t), z(t)]^{\mathsf{T}}$; (ii) computing the normalized points $\boldsymbol{p}(t)/\|\boldsymbol{p}(t)\|_2$; (iii) appending 50 "noise dimensions" to obtain vectors in $\mathbb{R}^{53}$; (iv) adding independent random noise to each coordinate of each point as drawn from the Gaussian distribution $\mathcal{N}(0, \eta^2)$; and lastly (v) re-normalizing each point to obtain vectors of length $m = 53$ lying on S^{m-1}. We performed classification on the resulting data samples. Note that the reason why we turned the original $\mathbb{R}^3$

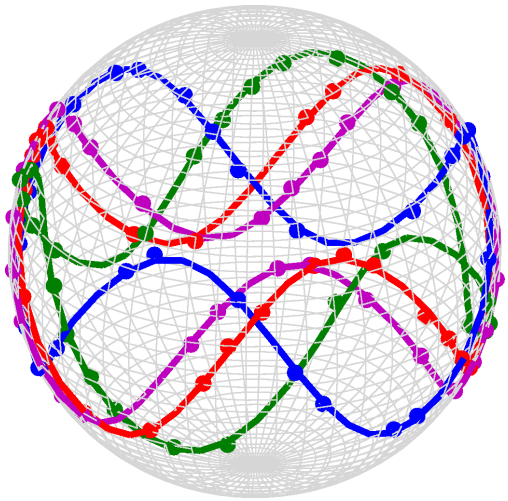


FIGURE 5.1. A realization of the first three coordinates of the synthetic database training set with $N_0 = 25$ and $\eta = 0.01$. Nodes denote training samples; colors denote classes.

problem into a problem in $\mathbb{R}^{53}$ was because SRC is designed for high-dimensional classification problems [**104**] and to make the problem more challenging. We emphasize that we did not apply any method of dimensionality reduction to this database.

Figure 5.1 shows the first three coordinates of a realization of the training set of the synthetic database. Note that the class manifold dimension is the same for each class and equal to 1. The signal-to-noise (SNR) ratios are displayed in Table 5.1 for $N_0 = 25$ and various values of noise level η. These results were obtained by averaging the mean training sample SNR over 100 realizations of the data set.

$\eta = 0.0001$	$\eta = 0.001$	$\eta = 0.005$	$\eta = 0.01$	$\eta = 0.015$	$\eta = 0.02$	$\eta = 0.03$	$\eta = 0.05$
62.85	42.84	28.86	22.86	19.35	16.89	13.45	9.25

TABLE 5.1. Mean training sample signal-to-noise ratio (in decibels) over 100 realizations of the synthetic database with $N_0 = 25$ and various values of noise level η.

5.3.2. Experimental Results. We performed experiments on this database, first varying the number of training samples in each class and then varying the amount of noise. The algorithms LPCA-SRC, SRC, $\text{SRC}_{\text{pruned}}$, and the TDC methods significantly exceeded LSDL-SRC and the kNN methods in terms of accuracy in these experiments. In particular, these latter three methods

were always outperformed by LPCA-SRC by at least 10% and often by as much as $30\% - 40\%$. Though kNN-Ext generally performed better than kNN, neither method was competitive due to its inability to distinguish individual class manifolds near intersections, a result of considering the classes in terms of a single sample (or tangent vector) at a time. On the other hand, LSDL-SRC was not *local enough*; despite its explicit locality term, this method was unable to distinguish the individual classes from within a local neighborhood of the test sample. Because of their poor performance, we do not report the results of these algorithms here.

In contrast, the approximations in LPCA-SRC, SRC, and $\text{SRC}_{\text{pruned}}$ often contained nonzero coefficients solely at one or two dictionary elements bordering the test sample (up to sign) on the correct class manifold. That is, these approximations were *very* sparse, and this sparsity often resulted in correct classification. The TDC methods, though generally not as competitive as these first three algorithms, also showed relatively good performance; when there was a large enough number of training samples in each class, the TDC class-specific subdictionaries were effective in discriminating between classes.

Figure 5.2 shows the average classification accuracy (over 100 trials) of the competitive algorithms as we varied the number of training samples in each class. We fixed the noise level $\eta = 0.001$. LPCA-SRC generally had the highest accuracy. On average, LPCA-SRC outperformed SRC by 3.5%, though this advantage slightly decreased as the sampling density increased and the tangent vectors became less useful, in the sense that there were often already enough nearby training samples in the ground truth class of $\boldsymbol{y}$ to accurately approximate it without the addition of tangent vectors. SRC and $\text{SRC}_{\text{pruned}}$ had comparable accuracy for all tried values of N_0, indicating that the pruning parameter r was effective in removing unnecessary training samples from the SRC dictionary. Further, the increased accuracy of LPCA-SRC over $\text{SRC}_{\text{pruned}}$ suggests that the tangent vectors in LPCA-SRC contributed meaningful class information.

The TDC methods performed relatively poorly for small values of N_0. At low sampling densities, the TDC subdictionaries were poor models of the (local) class manifolds, leading to approximations of $\boldsymbol{y}$ that were often indistinguishable from each other and resulting in poor classification. Both TDC methods improved significantly as N_0 increased, with TDC2 outperforming TDC1 and in fact becoming comparable to LPCA-SRC for $N_0 \geq 60$. We attribute this to the extremely local nature

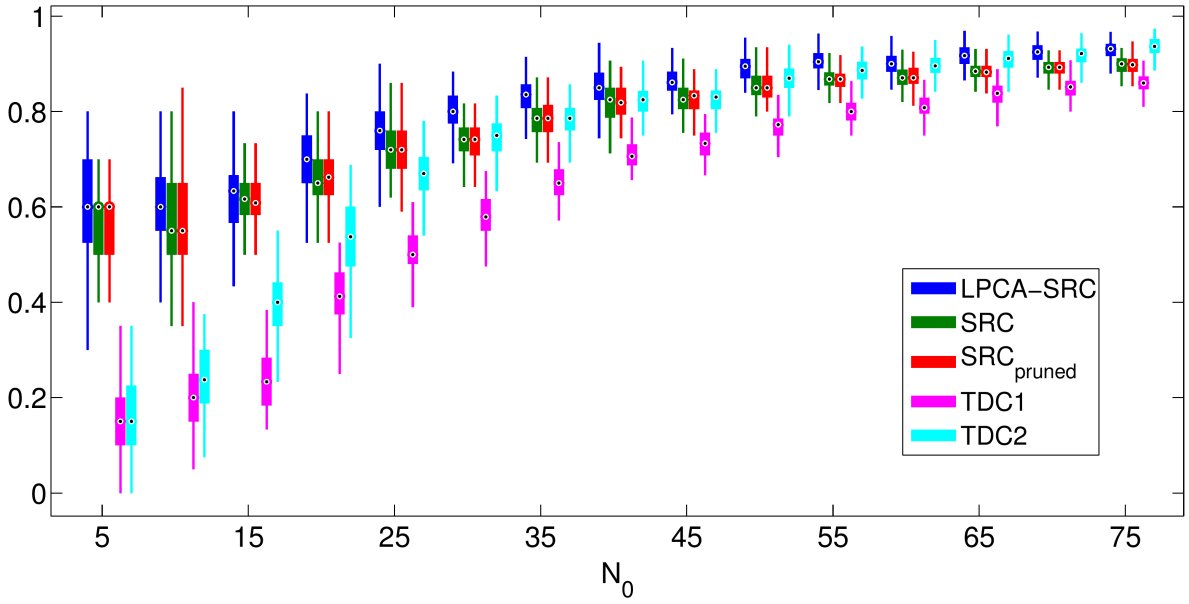


FIGURE 5.2. Box plot of the average classification accuracy (over 100 trials) of LPCA-SRC and competitive algorithms on the synthetic database with varying training class size N_0. We fixed $\eta = 0.001$.

of TDC2: It considers a single local patch on a class manifold at a time, rather than each class as a whole. Hence under dense sampling conditions, TDC2 effectively mimicked the successful use of sparsity in LPCA-SRC.

In Table 5.2, we display the runtime-related information of the competitive algorithms with varying training class size. In particular, we report the average runtime (in seconds), the number of columns in each algorithm's dictionary (we refer to this as the "size" of the dictionary, as the sample dimension is fixed), and the number of HOMOTOPY iterations. These latter variables are denoted N and κ, respectively. The runtime does not include the time it took to perform cross-validation. For the TDC methods, we report the average subdictionary sizes, and for conciseness, we display the results for only a handful of the values of N_0. We use "N/A" to indicate that a particular statistic is not applicable to the given algorithm.

The dictionary sizes of LPCA-SRC, SRC, and $\text{SRC}_{\text{pruned}}$ are quite informative. Recall that LPCA-SRC outperformed SRC and $\text{SRC}_{\text{pruned}}$ (by more than 3%) for the shown values of N_0. For $N_0 = 5$, the dictionary in LPCA-SRC was larger than that of the two other methods, adaptively

	$N_0 = 5$			$N_0 = 25$			$N_0 = 45$			$N_0 = 65$		
Algorithm	t	N	κ	t	N	κ	t	N	κ	t	N	κ
LPCA-SRC	0.01	56	2	0.07	80	3	0.12	42	3	0.16	30	2
SRC	0.00	20	2	0.04	100	3	0.10	180	3	0.16	260	3
$\text{SRC}_{\text{pruned}}$	0.01	20	2	0.05	79	3	0.13	146	3	0.21	201	3
TDC1	0.01	9	N/A	0.04	6	N/A	0.07	5	N/A	0.09	3	N/A
TDC2	0.02	3	N/A	0.06	2	N/A	0.09	2	N/A	0.13	2	N/A

TABLE 5.2. Average runtime in seconds (t), dictionary size (N), and number of HOMOTOPY iterations (κ) over 100 trials of LPCA-SRC and competitive algorithms on the synthetic database with varying training class size N_0. We fixed $\eta = 0.001$.

retaining more samples to counter-balance the low sampling density. At large values of N_0, LPCA-SRC took full advantage of the increased sampling density, stringently pruning the set of training samples and keeping only those very close to $\boldsymbol{y}$. Due to the resulting small dictionary, it had comparable runtime to SRC despite its additional cost of computing tangent vectors. In contrast, without the addition of tangent vectors, $\text{SRC}_{\text{pruned}}$ was forced to keep a large number of training samples in its dictionary; the cost of the dictionary pruning step resulted in $\text{SRC}_{\text{pruned}}$ running slower than SRC, despite its slightly smaller dictionary. (We note that one might expect that $\text{SRC}_{\text{pruned}}$ would always have a smaller dictionary than LPCA-SRC since it does not include tangent vectors; this is not the case, as the value of the number-of-neighbors parameter n, and hence the pruning parameter r, may be different for the two algorithms.)

The TDC methods ran relatively fast, especially for large values of N_0. This is expected, as these algorithms do not require ℓ^1-minimization.

Figure 5.3 shows the average classification accuracy (over 100 trials) of the competitive algorithms as we varied the amount of noise. We fixed $N_0 = 25$. (Note that we do not show the runtime results for these experiments; those for varying class size in Table 5.2 are much more revealing.) LPCA-SRC had the highest classification accuracy for low values of η (equivalently, when the SNR was high), outperforming SRC by as much as nearly 4%. For $\eta \geq 0.015$ (i.e., when the SNR dropped below 20 decibels), LPCA-SRC lost its advantage over SRC and $\text{SRC}_{\text{pruned}}$. This is likely due to noise degrading the accuracy of the tangent vectors. SRC and $\text{SRC}_{\text{pruned}}$ had nearly identical accuracy for all values of η; again, this illustrates that faraway training samples (as defined by the pruning parameter r) did not contribute to the ℓ^1-minimized approximation of the test sample, and the increased accuracy of LPCA-SRC over $\text{SRC}_{\text{pruned}}$ for low noise values demonstrates the

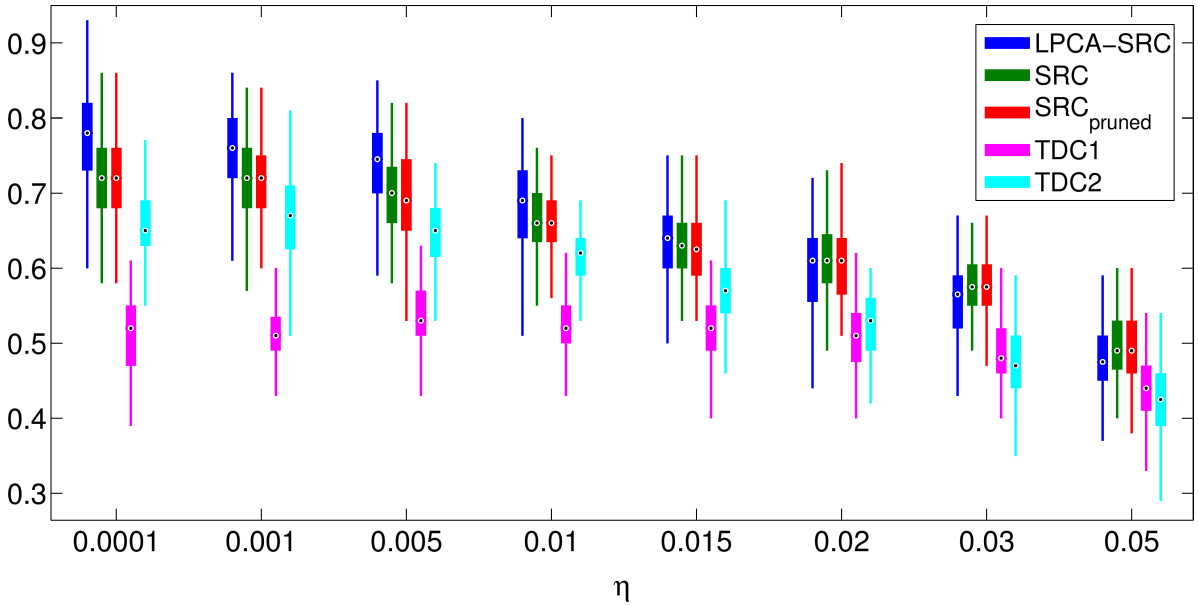


FIGURE 5.3. Box plot of the average classification accuracy (over 100 trials) of LPCA-SRC and competitive algorithms on the synthetic databases with varying noise level η. We fixed $N_0 = 25$.

efficacy of the tangent vectors in LPCA-SRC in these cases. We briefly note that when we vary the noise level for larger values of N_0, the accuracy of the tangent vectors generally improves. As a result, we see that LPCA-SRC can tolerate higher values of η before being outperformed by SRC and $\text{SRC}_{\text{pruned}}$.

TDC2 outperformed TDC1 for all but the largest values of η, though both algorithms were outperformed by the three SRC methods at this relatively low sampling density for the reasons discussed previously. For $\eta \geq 0.03$, TDC2 began performing worse than TDC1. We expect that the local patches represented by the subdictionaries in TDC2 became poor estimates of the (tangent hyperplanes of the) class manifolds as the noise increased, resulting in a decrease in classification accuracy.

In summary, the experimental results on the synthetic database show that LPCA-SRC can achieve higher classification accuracy than SRC and similar methods when the class manifolds are sparsely-sampled and the SNR is large. In these cases, the tangent vectors in LPCA-SRC help to "fill out" portions of the class manifolds that lack training samples. When the sampling density was sufficiently high, however, we saw that the tangent vectors in LPCA-SRC were less needed

to provide an accurate, local approximation of the test sample, and thus LPCA-SRC offered a smaller advantage over SRC and $\text{SRC}_{\text{pruned}}$. Additionally, for higher noise (i.e., low SNR) cases, the computed tangent vectors were less reliable and the classification performance consequently deteriorated. With regard to runtime, LPCA-SRC appeared to adapt to the sampling density of the synthetic database, and though the addition of tangent vectors initially increased the dictionary size in LPCA-SRC, the online dictionary pruning step allowed for runtime comparable to SRC when the class sizes were large.

5.4. Face Databases

5.4.1. Database Description. The *AR Face Database* [**65**] contains 70 male and 56 female subjects photographed in two separate sessions held on different days. Each session produced 13 images of each subject, the first seven with varying lighting conditions and expressions, and the remaining six images occluded by either sunglasses or scarves under varying lighting conditions. Images were cropped to 165×120 pixels and converted to grayscale. In our experiments, we selected the first 50 male subjects and first 50 female subjects, as was done in several papers (e.g., Wright et al. [**104**]), for a total of 100 classes. We performed classification on two versions of this database. The first, which we call "AR-1," contains the 1400 un-occluded images from both sessions. The second version, "AR-2," consists of the images in AR-1 as well as the 600 occluded images (sunglasses and scarves) from Session 1.

The *Extended Yale Face Database B* [**44**] contains 38 classes (subjects) with about 64 images per class. The subjects were photographed from the front under various lighting conditions. We used the version of Extended Yale B that contains manually-aligned, cropped, and resized images of dimension 192×168.

The *Database of Faces* (formerly "The ORL Database of Faces") [**2**] contains 40 classes (subjects) with 10 images per class. The subjects were photographed from the front against dark, homogeneous backgrounds. The sets of images of some subjects contain varying lighting conditions, expressions, and facial details. Each image in ORL is initially of 92×112 pixels.

Given existing work on the manifold structure of face databases (e.g., that of Saul and Roweis [**78**], He et al. [**50**], and Lee et al. [**59**]), we make the following suppositions: Since images in each

class in AR-1 and AR-2 have extreme variations in lighting conditions and differing expressions, the class manifolds of these databases may be nonlinear. Further, the natural occlusions contained in AR-2 make these class manifolds *highly* nonlinear. Alternatively, since the images in each class in Extended Yale B differ primarily in lighting conditions, the class manifolds may be nearly linear. Lastly, since the images in some classes in ORL differ in both lighting conditions and expression, these class manifolds may be nonlinear; however, since the variations are small, these manifolds may be well-approximated by linear subspaces.

With regard to sampling density, we reiterate that Extended Yale B has large class sizes compared to AR and ORL. In our experiments, we randomly selected the same number of samples in each class to use for training, i.e., we set $N_0 \equiv N_l$, $1 \leq l \leq L$, where N_0 was half the number of samples in each class.[1] We used the remaining samples for testing.

5.4.2. Dimensionality Reduction. To perform dimensionality reduction on the face databases, we used (global) PCA to transform the raw images to $m_{\text{PCA}} \in \{30, 56, 120\}$ dimensions before performing classification. Similar values for m_{PCA} were used by Wright et al. [**104**]. For the remainder of this chapter as well as Chapter 7, we will refer to the PCA-compressed versions of the raw face images as "feature vectors" and m_{PCA} as the "feature dimension." We note that the data was not centered (around the origin) in the PCA transform space.

5.4.3. Handling Occlusion. Since AR-2 contains images with occlusion, we considered using the "occlusion version" of SRC (with analogous modifications to LPCA-SRC and $\text{SRC}_{\text{pruned}}$) on this database. As discussed by Wright et al. [**104**], this model assumes that $\boldsymbol{y}$ is the summation of the (unknown) true test sample $\boldsymbol{y}_0$ and an (unknown) sparse error vector. The resulting modified ℓ^1-minimization problem consists of appending the dictionary of training samples with the identity matrix $I \in \mathbb{R}^{m \times m}$ and decomposing $\boldsymbol{y}$ over this augmented dictionary. For more details, see Section 3.2 of the SRC paper [**104**]. It can be shown that the occlusion version of SRC is equivalent to CRC (see Section 3.3.1) with $p = q = 1$.

However, the context in which Wright et al. use the occlusion version of SRC on the AR database is critically different than our experimental setup here [**104**]. In the SRC paper, the samples with occlusion make up the test set. In our case, both the training and test set contain samples with and

[1]Since the class sizes vary slightly in Extended Yale B, we set $N_0 = 32$ on this database.

without occlusion. As a consequence, occluded samples in the training set can be used to express test samples with occlusion, and on the other hand, the use of the identity matrix to extend the dictionary in SRC results in too much error allowed in the approximation of un-occluded samples. Correspondingly, we see much worse classification performance in SRC when we use its occlusion version on AR-2. Hence, we stick to Algorithm 1 (the original version of SRC) on all face databases.

5.4.4. AR Face Database Results. Table 5.3 displays the average accuracy and standard error over 10 trials for the two versions of AR. LPCA-SRC had substantially higher classification accuracy than the other methods on both versions of AR with $m_{\text{PCA}} = 30$. This suggests that the tangent vectors in LPCA-SRC were able to recover important class information lost in the stringent PCA dimensionality reduction. As m_{PCA} increased, however, the methods SRC, $\text{SRC}_{\text{pruned}}$, and LSDL-SRC became more competitive, as more discriminative information was retained in the feature vectors and less needed to be provided by the LPCA-SRC tangent vectors. $\text{SRC}_{\text{pruned}}$ had comparable accuracy to SRC, indicating that, once again, training samples could be removed from the SRC dictionary using the pruning parameter r without decreasing classification accuracy. In some cases, the removal of these faraway training samples slightly improved class discrimination.

For the most part, the other algorithms performed poorly on AR. The exception was LSDL-SRC, which had comparable accuracy to LPCA-SRC for $m_{\text{PCA}} = 120$ (slightly outperforming it for AR-1) and beat SRC on AR-1 for $m_{\text{PCA}} = 56$. However, LSDL-SRC had lower accuracy than the SRC algorithms for $m_{\text{PCA}} = 30$ on both versions of this database. In contrast, the TDC methods performed relatively better for $m_{\text{PCA}} = 30$ than for larger values of m_{PCA} due to their more effective use of tangent vectors at this small feature dimension. Overall, however, their class-specific dictionaries were not as effective on this nonlinear, sparsely-sampled database as the multi-class dictionaries of the previously-discussed algorithms. Further, TDC2 often had notably high standard error, presumably because of its sensitivity to the value of the manifold dimension estimate d. This could perhaps be mitigated by using a different cross-validation procedure. Lastly, kNN and kNN-Ext had the lowest classification accuracies, though kNN-Ext offered a slight improvement over kNN. Both methods consistently selected $k = 1$ during cross-validation.

	AR-1						AR-2					
	$m_{\text{PCA}} = 30$		$m_{\text{PCA}} = 56$		$m_{\text{PCA}} = 120$		$m_{\text{PCA}} = 30$		$m_{\text{PCA}} = 56$		$m_{\text{PCA}} = 120$	
Algorithm	Acc	SE	Acc	SE	Acc	SE	Acc	SE	Acc	SE	Acc	SE
LPCA-SRC	**0.8663**	4.1	**0.9544**	2.3	0.9711	1.7	**0.7328**	6.0	**0.8844**	3.6	**0.9512**	2.6
SRC	0.8273	4.2	0.9357	2.6	0.9631	1.6	0.6945	4.3	0.8713	2.0	0.9450	2.4
$\text{SRC}_{\text{pruned}}$	0.8277	4.8	0.9353	3.8	0.9651	1.8	0.7092	4.0	0.8781	2.7	0.9459	2.5
TDC1	0.8046	6.5	0.8430	5.1	0.8634	5.9	0.6899	4.1	0.7603	3.7	0.7985	4.5
TDC2	0.7549	19.4	0.8137	11.9	0.8303	15.4	0.6422	16.1	0.7386	3.3	0.7735	4.4
LSDL-SRC	0.8184	4.0	0.9424	2.0	**0.9756**	0.9	0.6585	5.6	0.8610	2.2	0.9498	2.9
kNN	0.5846	4.6	0.6301	8.0	0.6461	4.9	0.4100	3.0	0.4297	5.0	0.4554	3.2
kNN-Ext	0.6036	4.5	0.6487	8.2	0.6677	4.7	0.4311	3.7	0.4526	2.9	0.4794	5.7

TABLE 5.3. Average accuracy and standard error ($\times 10^{-3}$) of LPCA-SRC and compared algorithms over 10 trials on AR.

Table 5.4 displays the average runtime and related results (over 10 trials) of the various classification algorithms for both versions of AR. The "dictionary size" N for kNN and kNN-Ext refers to the average size of the set from which the k-nearest neighbors are selected (e.g., for kNN, $N = N_{\text{tr}}$).

The generally large dictionary sizes of LPCA-SRC (and its consequently long runtimes) indicate that minimal dictionary pruning often occurred. Thus LPCA-SRC was generally slower than SRC and $\text{SRC}_{\text{pruned}}$. However, on AR-2 with $m_{\text{PCA}} = 30$, LPCA-SRC was able to eliminate many training samples from its dictionary, due to its effective use of tangent vectors on the (presumably) highly-nonlinear class manifolds of AR-2. At this low feature dimension, the computed tangent vectors contained more class discriminative information than nonlocal training samples, likely allowing for a more accurate—and local—approximation of $\boldsymbol{y}$ on its ground truth class manifold. LPCA-SRC was faster than SRC and $\text{SRC}_{\text{pruned}}$ (which kept a large number of training samples) in this case, and this is impressive, considering that LPCA-SRC also outperformed these methods by nearly 4% and more than 2%, respectively.

Despite not requiring ℓ^1-minimization, the TDC methods were often the slowest algorithms on the AR databases. We suspect that this is largely due to the relatively large number of classes in AR—recall that both TDC methods must compute least squares solutions (in TDC2, sometimes many of them) for each class represented in the pruned dictionary $D_{\boldsymbol{y}}^{\text{TDC}}$. Further, TDC2 selected a relatively large value of d during cross-validation (presumably so that its subdictionaries would contain a wider "snapshot" of the class manifolds), which made it even less efficient. The runtime of LSDL-SRC, unlike those of most of the other algorithms, was fairly insensitive to the feature

dimension, and as a result, LSDL-SRC was relatively efficient for $m_{\text{PCA}} \in \{56, 120\}$. However, the expense of its dictionary learning phase for $m_{\text{PCA}} = 30$, at which the ℓ^1-minimization algorithm in the SRC methods could be solved efficiently, resulted in LSDL-SRC's relatively slow runtime. Both kNN methods ran significantly faster than all the other methods.

	AR-1								
	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	t	N	κ	t	N	κ	t	N	κ
LPCA-SRC	7.25	435	61	12.50	676	87	19.07	795	112
SRC	6.11	700	51	8.87	700	72	13.57	700	99
$\text{SRC}_{\text{pruned}}$	3.76	231	39	5.10	226	49	6.90	232	60
TDC1	11.82	16	N/A	14.24	16	N/A	24.30	19	N/A
TDC2	8.89	5	N/A	16.79	5	N/A	36.68	5	N/A
LSDL-SRC	7.78	440	N/A	8.55	470	N/A	9.72	490	N/A
kNN	0.01	700	N/A	0.01	700	N/A	0.02	700	N/A
kNN-Ext	0.08	2170	N/A	0.09	2240	N/A	0.16	2660	N/A
	AR-2								
	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	t	N	κ	t	N	κ	t	N	κ
LPCA-SRC	10.53	478	58	35.27	1593	10	56.17	1690	151
SRC	11.39	1000	58	17.67	1000	85	27.74	1000	121
$\text{SRC}_{\text{pruned}}$	11.12	788	54	16.63	775	77	24.88	767	107
TDC1	20.56	25	N/A	27.51	26	N/A	43.07	26	N/A
TDC2	20.93	6	N/A	47.57	6	N/A	103.80	6	N/A
LSDL-SRC	22.70	750	N/A	16.34	620	N/A	22.19	710	N/A
kNN	0.02	1000	N/A	0.02	1000	N/A	0.04	1000	N/A
kNN-Ext	0.13	4300	N/A	0.15	3600	N/A	0.29	4400	N/A

TABLE 5.4. Average runtime in seconds (t), dictionary size (N), and number of HOMOTOPY iterations (κ) over 10 trials of LPCA-SRC and compared algorithms on AR.

5.4.5. Extended Yale Face Database B and Database of Faces ("ORL") Results. Table 5.5 displays the average accuracy and standard error for Extended Yale B (over 10 trials) and ORL (over 50 trials). On Extended Yale B, LPCA-SRC had the highest accuracy for all m_{PCA}, though as we saw on the AR database, this advantage decreased as m_{PCA} increased and SRC became more competitive. SRC and $\text{SRC}_{\text{pruned}}$ had very similar accuracy, indicating that training samples excluded from the dictionary via the pruning parameter r did not provide class information in the SRC framework. TDC1 and TDC2 had consistently mediocre performance, neither one outperforming the other over all settings of m_{PCA}, and LSDL-SRC improved as m_{PCA} increased, analogous to its behavior on AR. However, LSDL-SRC was clearly outperformed by LPCA-SRC even for $m_{\text{PCA}} = 120$, suggesting that the improved approximations in LPCA-SRC via its use

of tangent vectors were more effective (even at this high feature dimension) than the procedure in LSDL-SRC.[2] Along these same lines, the tangent vectors in kNN-Ext offered a considerable improvement over kNN, though once again both methods reported lower accuracy than all the other algorithms. As on AR, the kNN methods consistently selected $k = 1$ during cross-validation.

On ORL, LPCA-SRC and $\text{SRC}_{\text{pruned}}$ had comparable accuracy and outperformed SRC. This indicates that: (i) the pruning parameter r in LPCA-SRC and $\text{SRC}_{\text{pruned}}$ was *helpful* to classification (instead of simply being benign); and (ii) the tangent vectors computed in LPCA-SRC were not. With regard to (i), it must be the case that faraway training samples—those in different classes from the test sample—contributed significantly to the approximation of the test sample in SRC, negatively affecting classification performance. This is an example of *sparsity not necessarily leading to locality* (as it is relevant to class discrimination), as discussed in the LSDL-SRC paper [**101**]. With regard to (ii), we suspect that the tangent vectors in LPCA-SRC were simply *unneeded* to improve the classification performance on ORL. Though the approximations in SRC contained nonzero coefficients at training samples not in the same class as $\boldsymbol{y}$—presumably because of the sparse sampling and nonlinear structure of the class manifolds—many of these wrong-class training samples could be eliminated simply based on their distance to $\boldsymbol{y}$. This suggests that ORL's class manifolds can be fairly well-separated via Euclidean distance. An additional reason for (ii) was because the PCA transform to the dimensions specified in this experiment did not result in a loss of too much information, at least compared to AR and Extended Yale B. See Table 5.8 at the end of Section 5.4.6 for this comparison.

All of the remaining methods performed relatively well on ORL. The accuracies of TDC1 and TDC2 were similar and comparable to those of SRC. We ascertained that the success of the TDC methods was not due to their use of tangent vectors but instead the result of their "per-class" approximations of the test sample. This approach was very effective on the (presumably) well-separated class manifolds of ORL. Strikingly, the accuracy of LSDL-SRC was relatively low for $m_{\text{PCA}} = 120$, opposite to the trend we saw on the previous face databases. The performance of LSDL-SRC could be improved for $m_{\text{PCA}} = 120$ on this database if the samples were centered (around the origin) after PCA dimensionality reduction. However, we confirmed that LDSL-SRC

[2]LSDL-SRC was also outperformed by SRC on Extended Yale B in our experiments, in contrast to the experiments by Wei et al. in which $m_{\text{PCA}} = 300$ [**101**].

was still outperformed by LPCA-SRC in this case (albeit by a smaller margin), and its performance with centering on the other face databases was much worse than our reported results. In contrast to the results on Extended Yale B, kNN-Ext only provided a slight increase in accuracy over kNN, with the tangent vectors mimicking their unnecessary role in LPCA-SRC on this database. The value $k = 1$ was consistently selected by both kNN and kNN-Ext during cross-validation.

	Extended Yale B						ORL					
	$m_{\text{PCA}} = 30$		$m_{\text{PCA}} = 56$		$m_{\text{PCA}} = 120$		$m_{\text{PCA}} = 30$		$m_{\text{PCA}} = 56$		$m_{\text{PCA}} = 120$	
Algorithm	Acc	SE	Acc	SE	Acc	SE	Acc	SE	Acc	SE	Acc	SE
LPCA-SRC	**0.9049**	2.9	**0.9530**	1.7	**0.9710**	1.6	**0.9507**	24.0	**0.9600**	18.0	0.9602	17.0
SRC	0.8803	2.6	0.9371	2.8	0.9633	1.4	0.9374	24.0	0.9437	22.9	0.9422	18.8
$\text{SRC}_{\text{pruned}}$	0.8804	2.7	0.9371	2.6	0.9635	1.5	0.9506	23.7	0.9580	23.5	**0.9605**	19.3
TDC1	0.8568	10.0	0.9285	2.0	0.9446	2.8	0.9364	27.1	0.9457	25.4	0.9455	21.1
TDC2	0.8826	3.9	0.9093	2.8	0.9283	3.5	0.9351	29.8	0.9429	31.1	0.9418	23.3
LSDL-SRC	0.7495	4.8	0.8774	2.5	0.9492	2.0	0.9358	25.2	0.9515	19.9	0.9251	24.3
kNN	0.4300	3.5	0.5346	2.6	0.6245	3.8	0.9332	26.3	0.9387	24.7	0.9396	23.2
kNN-Ext	0.5464	6.9	0.6321	5.6	0.7058	5.4	0.9338	30.5	0.9412	28.9	0.9386	23.9

TABLE 5.5. Average accuracy and standard error ($\times 10^{-4}$) of LPCA-SRC and compared algorithms on Extended Yale B (over 10 trials) and ORL (over 50 trials).

Tables 5.6 and 5.7 show the runtime and related results for the Extended Yale B and ORL experiments, respectively. LPCA-SRC had much longer runtimes than SRC on Extended Yale B, especially as m_{PCA} increased. This was due to a combination of large values for d selected during cross-validation and the tangent vectors' decreasing efficacy at larger feature dimensions. However, the dictionary pruning procedure in LPCA-SRC actually eliminated a large number of training samples for all m_{PCA}; once again, the computed tangent vectors contained more class-discriminating information than the eliminated nonlocal training samples, especially at lower feature dimensions for which details provided by these tangent vectors were especially needed. The (presumed) linearity of the class manifolds of Extended Yale B, combined with this database's relatively dense sampling, lent itself well to the accurate computation of tangent vectors—part of the reason why LPCA-SRC used so many of them. Viewing these points as newly-generated and nearby training samples, LPCA-SRC's boost in accuracy over SRC can be viewed as an argument for locality in classification. We note that we might be able to decrease the value of d in LPCA-SRC while still maintaining an advantage over SRC (see the discussion in Section 4.4.1); our cross-validation procedure is designed to obtain the highest accuracy with no regard to computational cost.

On Extended Yale B, the TDC methods ran relatively more quickly (compared to the other algorithms) than on AR, presumably due to the much smaller number of classes on this database; both had runtimes typically between those of LPCA-SRC and SRC. Again, we see that LSDL-SRC had a relatively slow runtime for $m_{\text{PCA}} = 30$ and became more competitive as m_{PCA} increased. Though both kNN and kNN-Ext were very fast, the large "dictionary sizes" in kNN-Ext made this algorithm clearly the slower of the two methods.

On ORL, LPCA-SRC and SRC had comparable runtimes, a result of rigorous dictionary pruning in LPCA-SRC. This algorithm and $\text{SRC}_{\text{pruned}}$ retained roughly the same number of training samples in their respective dictionaries, and the latter was notably fast, running in about half the time as SRC. The remaining algorithms were even more efficient. TDC1 and TDC2 had comparable runtimes, both running faster than LSDL-SRC. As before, kNN and kNN-Ext had the fastest runtimes; the former was faster than the latter.

	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	t	N	κ	t	N	κ	t	N	κ
LPCA-SRC	29.20	1922	75	72.12	3359	120	141.97	3785	182
SRC	15.58	1216	62	24.70	1216	91	41.94	1216	137
$\text{SRC}_{\text{pruned}}$	15.92	1111	61	23.81	1112	88	40.50	1115	131
TDC1	8.10	20	N/A	27.62	59	N/A	42.83	59	N/A
TDC2	11.68	6	N/A	23.51	6	N/A	56.01	6	N/A
LSDL-SRC	67.30	1186	N/A	53.03	1003	N/A	38.73	821	N/A
kNN	0.02	1216	N/A	0.03	1216	N/A	0.05	1216	N/A
kNN-Ext	0.17	5350	N/A	0.25	4742	N/A	0.44	4864	N/A

TABLE 5.6. Average runtime in seconds (t), dictionary size (N), and number of HOMOTOPY iterations (κ) over 10 trials of LPCA-SRC and compared algorithms on Extended Yale B.

5.4.6. Tangent Vectors and PCA Feature Dimension. In this section, we offer evidence to support our claim that the tangent vectors in LPCA-SRC can recover discriminative information lost during PCA transforms to low dimensions. Thus LPCA-SRC can offer a clear advantage over SRC in these cases, as we saw in experimental results on AR and Extended Yale B.

In Figures 5.4-5.6, we display three versions of three example images from AR-1. The first version is the original image (before PCA dimensionality reduction), the second version is the recovered image from PCA dimensionality reduction to dimension $m_{\text{PCA}} = 30$, and the third version is the recovered corresponding tangent vector computed in LPCA-SRC. In each case, the

	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	t	N	κ	t	N	κ	t	N	κ
LPCA-SRC	0.54	59	26	0.73	72	34	1.22	111	50
SRC	0.85	200	40	1.34	200	57	2.09	200	81
$\text{SRC}_{\text{pruned}}$	0.25	19	12	0.34	26	16	0.53	39	24
TDC1	0.12	1	N/A	0.16	1	N/A	0.34	1	N/A
TDC2	0.12	3	N/A	0.23	3	N/A	0.53	3	N/A
LSDL-SRC	1.04	116	N/A	1.09	121	N/A	0.93	102	N/A
kNN	0.01	200	N/A	0.01	200	N/A	0.01	200	N/A
kNN-Ext	0.03	568	N/A	0.03	592	N/A	0.04	568	N/A

TABLE 5.7. Average runtime in seconds (t), dictionary size (N), and number of HOMOTOPY iterations (κ) over 50 trials of LPCA-SRC and compared algorithms on ORL.

tangent vector contains details of the original image not found in the recovered image, supporting our claim that the tangent vectors in LPCA-SRC can recover some (but not all) of the information lost in stringent PCA dimensionality reduction.

Towards quantifying what we mean by "stringent," Table 5.8 lists the average energy (over 10 trials) retained in the first m_{PCA} left-singular vectors of the face database training sets, along with the percent improvement in the accuracy of LPCA-SRC with respect to that of SRC and $\text{SRC}_{\text{pruned}}$. We reiterate that the addition of tangent vectors did not increase classification accuracy on ORL. Taking this into account, we see a correlation between the efficacy of tangent vectors in LPCA-SRC and the stringency of the PCA dimensionality reduction.

(a) Original Image (b) Recovered Image (c) Tangent Vector

FIGURE 5.4. The tangent vector does a much better job of displaying facial details conveying "happiness" (displayed in the original image) than the recovered image. Images (b) and (c) were recovered from PCA dimension $m_{\text{PCA}} = 30$.

(a) Original Image (b) Recovered Image (c) Tangent Vector

FIGURE 5.5. The tangent vector does a better job of displaying facial details conveying "anger" (displayed in the original image) than the recovered image, most notably in the subject's eyes and eyebrows. Images (b) and (c) were recovered from PCA dimension $m_{\text{PCA}} = 30$.

(a) Original Image (b) Recovered Image (c) Tangent Vector

FIGURE 5.6. The tangent vector shows the subject's smile, in particular, its shape, shading, and teeth and cheek detail (notice the right dimple), better than the recovered image. Images (b) and (c) were recovered from PCA dimension $m_{\text{PCA}} = 30$.

5.4.7. Summary. The experimental results on face databases show that LPCA-SRC can achieve higher accuracy than SRC in cases of low sampling and/or nonlinear class manifolds and small PCA feature dimension. We showed that LPCA-SRC had a significant advantage over SRC and the other algorithms for the small class sizes and nonlinear class manifolds of the AR database when the feature dimension was low. We also showed that LPCA-SRC could improve classification

	$m_{\text{PCA}} = 30$		$m_{\text{PCA}} = 56$		$m_{\text{PCA}} = 120$	
Database	Energy	% Increased Acc. SRC / $\text{SRC}_{\text{pruned}}$	Energy	% Increased Acc. SRC / $\text{SRC}_{\text{pruned}}$	Energy	% Increased Acc. SRC / $\text{SRC}_{\text{pruned}}$
AR-1	0.4527	3.90/3.86	0.5322	1.87/1.91	0.6522	0.80/0.60
AR-2	0.4137	3.83/2.36	0.4884	1.31/0.63	0.5988	0.62/0.53
Extended Yale B	0.3954	2.46/2.45	0.4803	1.59/1.59	0.6055	0.77/0.74
ORL	0.5385	1.34/0.05	0.6581	1.26/-0.04	0.8487	1.73/0.03

TABLE 5.8. Average energy retained in PCA dimensionality reduction (over 10 trials) to various dimensions m_{PCA} for the face database training sets, as well as the average increase in classification accuracy of LPCA-SRC over SRC and $\text{SRC}_{\text{pruned}}$.

on Extended Yale B and ORL through its use of tangent vectors to provide a local approximation of the test sample and its discriminating pruning parameter, respectively.

The runtime of LPCA-SRC was sometimes much longer than that of SRC, although this was less often seen for small feature dimensions, at which LPCA-SRC tended to excel. The size of the dictionary in LPCA-SRC was observed to be a good predictor of the relationship between the runtimes of LPCA-SRC and SRC, and this could easily be computed (given estimates of the parameters n and d) before deciding between the two methods.

To validate our claim that the tangent vectors in LPCA-SRC can contain information lost in stringent PCA dimensionality reduction, we provided examples from the AR database. We also compared the energy retained in PCA dimensionality reduction with the increase in accuracy in LPCA-SRC over SRC and saw that they were inversely correlated.

CHAPTER 6

Bounding the Tangent Error

In this chapter, we aim to bound the distance between a tangent vector computed in LPCA-SRC and the corresponding class manifold.

6.1. Description and Assumptions

Let $\mathcal{M}$ be a smooth d-dimensional manifold in $\mathbb{R}^m$, and let $\boldsymbol{x}_0$ be a point on $\mathcal{M}$. Suppose that we use local PCA to compute the d basis vectors of the approximated tangent hyperplane of $\mathcal{M}$ at $\boldsymbol{x}_0$. Since shifted (by $\boldsymbol{x}_0$) and scaled versions of these basis vectors lie approximately on $\mathcal{M}$, this technique may be used to increase the sampling density around $\boldsymbol{x}_0$. For classification purposes, this can be used to increase the size of the training set and give us a better picture of each class manifold, as in our proposed algorithm, LPCA-SRC.

However, this approach may lead to bad training sample estimates in the sense that the tangent vectors are far from the original manifold. Local PCA may give us a poor approximation of the true tangent hyperplane under factors such as significant noise or poor sampling. It is also possible that our scaling factor produces points that are too far from $\boldsymbol{x}_0$ given the local curvature.

In what follows, we assume that the scale factor $c \geq 0$ is fixed. (Recall that $c := r\gamma$ for $\gamma \sim \text{unif}(0, 1)$ in LPCA-SRC.) We aim to bound, with high probability, the distance between an arbitrary LPCA-SRC tangent vector (defined as a shifted and scaled tangent hyperplane basis vector as computed using local PCA) and its closest point on $\mathcal{M}$ for the case that $d = 1$. Our work uses a (modified) result of Kaslovsky and Meyer [**57**]. The bound we produce is stated in terms of the principal curvatures of $\mathcal{M}$ at $\boldsymbol{x}_0$, the number and noise level of the points sampled from $\mathcal{M}$ and used in local PCA, and the ambient dimension m. It also requires certain continuity and differentiability assumptions on $\mathcal{M}$.

Our work in this chapter also assumes that the local tangent hyperplane origin $\boldsymbol{x}_0$ lies on the manifold $\mathcal{M}$. In the context of LPCA-SRC, this is not always the case, as training samples—where

the tangent vectors are constructed—are likely to contain noise. However, a step can be added to the offline phase of the LPCA-SRC algorithm to ensure that each local origin lies approximately on its class manifold. See Section 6.5 for details regarding this procedure.

6.2. Kaslovsky and Meyer's Tangent Bound

Kaslovsky and Meyer [**57**] computed a bound on the angle between the true tangent hyperplane and the approximated tangent hyperplane as found by local PCA. This bound holds with high probability. Let $P := UU^\mathsf{T}$ be the orthogonal projector onto the true tangent space L, and let $\widehat{P} := \widehat{U}\widehat{U}^\mathsf{T}$ be the orthogonal projector onto the approximated tangent space $\widehat{L}$. Here, the columns of U and $\widehat{U}$ are orthonormal bases of the true and approximated tangent (hyper)planes, respectively. U is unknown, but $\widehat{U}$ is found via local PCA, i.e., its columns are the first d eigenvectors of the sample covariance matrix constructed from local samples. Kaslovsky and Meyer [**57**] bounded (with high probability) the quantity $\|P - \widehat{P}\|_F$ in terms of (i) the manifold dimension d; (ii) the ambient dimension m; (iii) the number of points n in the sample covariance matrix, their noise level σ, and r, their maximum distance from the the point $\boldsymbol{x}_0$ as measured along the (true) tangent plane; and (iv) the principal curvatures of $\mathcal{M}$ at $\boldsymbol{x}_0$. The bound also depends on two probability constants, ξ and ξ_λ, which can be tuned so that the probability that the bound holds is sufficiently high.

There are two conditions for the bound holding. The first, roughly, is the requirement that the true tangent space is sufficiently separable from the sample noise σ and the curvature of $\mathcal{M}$ at $\boldsymbol{x}_0$ at some scale (or neighborhood size) derived from n [**57**]. The second condition is a technical assumption, which, given that the first condition holds, will be satisfied if the sampling density is high enough [**57**]. Additionally, there are some "benign assumptions" on n and the probability constants ξ and ξ_λ [**57**]. We can write the Kaslovsky and Meyer bound as

$$\|P - \widehat{P}\|_F \leq E(d, m, n, r, K, \sigma, \xi, \xi_\lambda),$$

where K is the $d \times (m - d)$ matrix of principal curvatures of $\mathcal{M}$ at $\boldsymbol{x}_0$.

Recall that in LPCA-SRC we use the version of local PCA by Singer and Wu [**87**], in particular, our implementation in Algorithm 4. Thus Kaslovsky and Meyer's result, derived in the case that standard local PCA is used, must be modified to hold in the Singer and Wu case. We do this in

Appendix A. Let us denote the modified bound (which is formally stated in Theorem A.6.2) by

$$\|P - \widehat{P}\|_F \leq E^*(d, m, n, r, K, \sigma, \xi, \xi_\lambda) =: E^*. \tag{6.1}$$

To give the reader a sense of the modified bound E^*, we can informally write

$$\|P - \widehat{P}\|_F \leq E^* \approx \frac{2\sqrt{2}}{\sqrt{n}} \frac{\left[K^{(+)} r^3 + \sigma\sqrt{d(m-d)}\left(\sigma + \frac{r}{\sqrt{d+2}} + \frac{\mathcal{K}_W^{1/2} r^2}{2\sqrt{(d+2)(d+4)}}\right)\right]}{\frac{r^2}{d+2} - \frac{\mathcal{K}_W r^4}{4(d+2)(d+4)} - \sigma^2(\sqrt{d} + \sqrt{m-d})},$$

where $\mathcal{K}_W$ and $K^{(+)}$ are curvature constants derived from the entries of the curvature matrix K.

6.3. Main Theorem

Notation: Let $\boldsymbol{x}_0$ be a point on the 1-dimensional manifold $\mathcal{M}$. Suppose that $\boldsymbol{u}$ is a unit basis vector of the tangent line at $\boldsymbol{x}_0$ on $\mathcal{M}$, and let $\hat{\boldsymbol{u}}$ be the approximation of $\boldsymbol{u}$ as found by Singer and Wu's local PCA [**87**]. For a constant $c \geq 0$, define $\hat{\boldsymbol{v}} := c\hat{\boldsymbol{u}}$, and assume without loss of generality that $\langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle \geq 0$. Lastly define the (shifted and scaled) computed tangent vector $\hat{\boldsymbol{z}} := \boldsymbol{x}_0 + \hat{\boldsymbol{v}}$.

THEOREM 6.3.1. *Suppose that $\mathcal{M}$ can be described by a curve $\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^m$, $m \geq 2$, with domain $\mathcal{D} \subset \mathbb{R}$ and $\boldsymbol{x}_0 = \boldsymbol{f}(t_0)$ for some $t_0 \in \mathcal{D}$, such that $\boldsymbol{f}$ satisfies the following conditions:*

(1) The principal directions of $\boldsymbol{f}$ at t_0, corresponding to the $m-1$ principal curvatures, exist;

(2) There exists $\epsilon > 0$ such that in a neighborhood $\mathcal{N}_\epsilon(\boldsymbol{x}_0) := \{\boldsymbol{f}(t) : \|\boldsymbol{f}(t) - \boldsymbol{f}(t_0)\|_2 < \epsilon\}$ of $\boldsymbol{x}_0$, there is a one-to-one mapping between points in $\mathcal{N}_\epsilon(\boldsymbol{x}_0)$ and the tangent line of $\mathcal{M}$ at $\boldsymbol{x}_0$. Further, $\boldsymbol{f}$ is parametrized so that $I_\epsilon(\boldsymbol{x}_0) := \{t \in \mathcal{D} : \boldsymbol{f}(t) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)\}$ is an interval on which $\boldsymbol{f}$ is continuous and one-to-one;

(3) The scaling factor c is small enough so that there exists $\boldsymbol{f}(t_c) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)$ that is mapped to the point $\boldsymbol{x}_0 + c\boldsymbol{u}$ on the tangent line via the one-to-one mapping in (2);

(4) $\boldsymbol{f}$ is three times differentiable on $I_\epsilon(\boldsymbol{x}_0)$ with continuous second-order derivatives in $\overline{I_\epsilon(\boldsymbol{x}_0)}$, and the third derivative of $\boldsymbol{f}$ satisfies $\sup_{\boldsymbol{f}(t) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)} \|\boldsymbol{f}'''(t)\|_2 =: M < \infty$.

*Suppose that all the conditions of the modified Kaslovsky and Meyer tangent bound [**57**] hold so that $\|P - \widehat{P}\|_F \leq E^*$ with probability greater than $p := 1 - 2e^{-\xi_\lambda^2} - 9e^{-\xi^2}$. Then the distance*

between the shifted and scaled computed tangent vector $\hat{\boldsymbol{z}}$ and the manifold $\mathcal{M}$ satisfies

$$\operatorname{dist}(\hat{\boldsymbol{z}} - \mathcal{M}) := \inf_{z' \in \mathcal{M}} \|\hat{\boldsymbol{z}} - \boldsymbol{z}'\|_2 \leq c\sqrt{\frac{E^*}{2}} + \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2} =: \delta \tag{6.2}$$

with probability greater than p, where

$$|\mathcal{R}| \leq R_{\text{bound}} := \frac{Mc^5}{6}(m-1)\Big(\kappa + \frac{Mc}{6}\Big).$$

Here, κ is given by

$$\kappa := \max_{i \in \{2,\ldots,m\}} |\kappa^{(i)}|,$$

where $\kappa^{(i)}$ is the principal curvature in the ith normal direction of $\mathcal{M}$ at $\boldsymbol{x}_0$, $i \in \{2,\ldots,m\}$.

Set $\boldsymbol{\kappa} = [\kappa^{(2)}, \ldots \kappa^{(m)}]^{\mathsf{T}} \in \mathbb{R}^{m-1}$ and define $\delta_{\text{approx}} := c\sqrt{\frac{E^*}{2}} + \frac{c^2}{2}\|\boldsymbol{\kappa}\|_2$ and

$$\zeta_{\text{bound}} := \max\left\{\left(\frac{c^4}{4}\|\boldsymbol{\kappa}\|_2^2 + R_{\text{bound}}\right)^{1/2} - \frac{c^2}{2}\|\boldsymbol{\kappa}\|_2, \frac{c^2}{2}\|\boldsymbol{\kappa}\|_2 - \left(\frac{c^4}{4}\|\boldsymbol{\kappa}\|_2^2 - R_{\text{bound}}\right)^{1/2}\right\}$$

Then δ_{approx} approximates the upper bound $\delta \geq \operatorname{dist}(\hat{\boldsymbol{z}} - \mathcal{M})$ with $|\delta - \delta_{\text{approx}}| \leq \zeta_{\text{bound}}$ with probability greater than p.

6.4. Proof of Main Theorem

We prove the main part of Theorem 6.3.1 using two lemmas, each of which essentially restates half the theorem.

6.4.1. First Lemma.

LEMMA 6.4.1. *Let $\mathcal{M}$ be a 1-dimensional manifold in $\mathbb{R}^m$ ($m \geq 2$) with $\boldsymbol{x}_0 \in \mathcal{M}$, and suppose that $\boldsymbol{u}$ is a unit basis vector of the tangent line at $\boldsymbol{x}_0$ on $\mathcal{M}$. For fixed $c \geq 0$, obtain a scaled and shifted version of $\boldsymbol{u}$ by setting $\boldsymbol{z} := \boldsymbol{x}_0 + c\boldsymbol{u}$. Suppose that $\mathcal{M}$ can be described by a curve $\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^m$, with domain $\mathcal{D} \subset \mathbb{R}$ and $\boldsymbol{x}_0 = \boldsymbol{f}(t_0)$ for some $t_0 \in \mathcal{D}$, such that the properties (1)-(4) in Theorem 6.3.1 hold. Then the distance between $\boldsymbol{z}$ and $\mathcal{M}$ satisfies*

$$\operatorname{dist}(\boldsymbol{z} - \mathcal{M}) \leq \frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R},$$

where

$$|\mathcal{R}| \leq \frac{Mc^5}{6}(m-1)\Big(\kappa + \frac{Mc}{6}\Big).$$

Here, κ is given by

$$\kappa := \max_{i \in \{2,\ldots,m\}} |\kappa^{(i)}|,$$

where $\kappa^{(i)}$ is the principal curvature in the ith normal direction of $\mathcal{M}$ at $\boldsymbol{x}_0$, $i \in \{2,\ldots,m\}$.

Proof of Lemma 6.4.1. The proof of this lemma relies heavily on the geometric data model in the work of Kaslovsky and Meyer [**57**] (see also the work of Tyagi et al. [**94**]).

Geometric setup in three dimensions. Let $m = 3$. Suppose that $\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^3$ is a smooth curve with coordinate functions $f_i : \mathbb{R} \to \mathbb{R}$, $i = 1, 2, 3$, and point on the curve

$$\boldsymbol{f}(t_0) = [f_1(t_0), f_2(t_0), f_3(t_0)]^\mathsf{T} =: \boldsymbol{x}_0.$$

By the assumptions in Lemma 6.4.1, we can compute the unit tangent vector $\boldsymbol{t}_{t_0}$, the unit normal vector $\boldsymbol{n}_{t_0}$, and the unit binormal vector $\boldsymbol{b}_{t_0}$ at t_0 on $\boldsymbol{f}$. By definition, all three vectors are mutually orthogonal. Further, $\boldsymbol{n}_{t_0}$ indicates the normal direction of curvature, and similarly, $\boldsymbol{b}_{t_0}$ indicates the normal direction of torsion, or second curvature.

Let R be the rotation matrix that maps $\boldsymbol{n}_{t_0}$ to $\boldsymbol{e}_2 := [0, 1, 0]^\mathsf{T}$ and $\boldsymbol{b}_{t_0}$ to $\boldsymbol{e}_3 := [0, 0, 1]^\mathsf{T}$. We define the function $\tilde{\boldsymbol{f}} := R(\boldsymbol{f} - \boldsymbol{x}_0)$. This transformation shifts $\boldsymbol{x}_0$ to the origin and aligns the normal directions of curvature (i.e., the principal directions) of $\boldsymbol{f}$ at t_0 with the last two coordinate axes. As a result of the orthogonality relationships, the unit tangent vector becomes aligned with the x-axis, and since $\boldsymbol{b}_{t_0} := \boldsymbol{t}_{t_0} \times \boldsymbol{n}_{t_0}$, $R(\boldsymbol{t}_{t_0}) = \boldsymbol{e}_1 := [1, 0, 0]^\mathsf{T}$. The function $\tilde{\boldsymbol{f}}$ can subsequently be parametrized locally by $\tilde{\boldsymbol{f}}(x) = [x, \tilde{f}_2(x), \tilde{f}_3(x)]^\mathsf{T}$, where $\tilde{f}_i : \mathbb{R} \to \mathbb{R}$, $i = 2, 3$ [**69**, **94**], as follows:

By the assumptions in Lemma 6.4.1, there exists ϵ such that each point $\boldsymbol{f}(t) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)$ has a unique mapping onto the tangent line at $\boldsymbol{x}_0$, and so there exists a bijective function $\gamma : I_\epsilon(\boldsymbol{x}_0) \to [-\tilde{\epsilon}, \tilde{\epsilon}]$ for some $0 < \tilde{\epsilon} \leq \epsilon$ (since we assumed that there is exactly one point $\boldsymbol{f}(t) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)$ for each $t \in I_\epsilon(\boldsymbol{x}_0)$). Further, since $\boldsymbol{f}$ is continuous on the interval $I_\epsilon(\boldsymbol{x}_0)$, we can define $x := \gamma(t) = \pm(t - t_0)$,

which renders the desired parametrization of $\tilde{\boldsymbol{f}}$ as

$$\tilde{\boldsymbol{f}}(x) = \tilde{\boldsymbol{f}}(\gamma(t)) := R(\boldsymbol{f}(t) - \boldsymbol{x}_0),$$

for $\boldsymbol{f}(t)$ close to $\boldsymbol{f}(t_0)$. It follows that the point $\boldsymbol{f}(t_0) = \boldsymbol{x}_0$ corresponds to $\tilde{\boldsymbol{f}}(0) = \mathbf{0}$ via

$$\mathbf{0} = R(0) = R(\boldsymbol{f}(t_0) - \boldsymbol{x}_0) = \tilde{\boldsymbol{f}}(\gamma(t_0)) = \tilde{\boldsymbol{f}}(\pm(t_0 - t_0)) = \tilde{\boldsymbol{f}}(0).$$

To further understand this correspondence, let $\boldsymbol{z} \in \mathbb{R}^3$ be a point on the tangent line of $\boldsymbol{f}$ at t_0 such that $\|\boldsymbol{z} - \boldsymbol{x}_0\|_2 = c$, for small $c \geq 0$ (i.e., $c \leq \tilde{\epsilon}$). There are two such points, and they can be differentiated by setting $\boldsymbol{z}_+$ to be the point in the direction of the unit tangent vector $\boldsymbol{t}_{t_0}$ and $\boldsymbol{z}_-$ to be the point in the opposite direction. Then $\boldsymbol{z}_+$ corresponds to $[c, 0, 0]^\mathsf{T}$ and $\boldsymbol{z}_-$ corresponds to $[-c, 0, 0]^\mathsf{T}$ in the shifted and rotated coordinate system. Further, there is a unique point $\boldsymbol{x}_{\boldsymbol{z}_+} = \boldsymbol{f}(t_{\boldsymbol{z}_+})$ in the original coordinate system that corresponds to $\tilde{\boldsymbol{f}}(c)$, where $c = \gamma(t_{\boldsymbol{z}_+}) = \pm(t_{\boldsymbol{z}_+} - t_0)$ and $\boldsymbol{f}(t_{\boldsymbol{z}_+})$ is close to $\boldsymbol{f}(t_0)$ (i.e., $\|\boldsymbol{f}(t_{\boldsymbol{z}_+}) - \boldsymbol{f}(t)\|_2 \leq \epsilon$). The precise correspondence is

$$\tilde{\boldsymbol{f}}(c) = \tilde{\boldsymbol{f}}(\gamma(t_{\boldsymbol{z}_+})) = R(\boldsymbol{f}(t_{\boldsymbol{z}_+}) - \boldsymbol{x}_0) = R(\boldsymbol{x}_{\boldsymbol{z}_+} - \boldsymbol{x}_0).$$

An analogous statement holds for $\boldsymbol{z}_-$.

The advantages of viewing points in the shifted and rotated coordinate system (e.g., $[c, 0, 0]^\mathsf{T}$ instead of $\boldsymbol{z}_+$) are (i) the relationship between the function and the tangent line are preserved, so that we may use $\tilde{\boldsymbol{f}}$ instead of $\boldsymbol{f}$ without loss of generality to measure the distance between a point near $\boldsymbol{x}_0$ on the tangent line and the curve $\boldsymbol{f}$, and (ii) the first nonzero terms in the Taylor expansion of $\tilde{f}_2$ and $\tilde{f}_3$ at 0 are quadratic:

$$\begin{aligned}\tilde{f}_i(x) &= \frac{1}{2}\tilde{f}_i''(x)\,x^2 + \text{ higher-order terms} \\ &= \frac{1}{2}\kappa^{(i)}x^2 + \text{ higher-order terms,}\end{aligned}$$

for $i = 2, 3$ [**45**]. Here, $\tilde{f}_i''$ denotes the second derivative of $\tilde{f}_i$, $i = 2, 3$. Additionally, $\kappa^{(2)}$ is the curvature of $\tilde{\boldsymbol{f}}$ at 0, and $\kappa^{(3)}$ is the torsion of $\tilde{\boldsymbol{f}}$ at 0; equivalently, $\kappa^{(i)}$ is the curvature in the ith

normal direction, $i = 2, 3$. Since curvature is invariant under rigid transformation, these values can be obtained from the original function $\boldsymbol{f}$ at t_0. More likely, however, since an explicit expression for $\boldsymbol{f}$ may be difficult or impossible to obtain in practice, these values can be estimated directly from $\mathcal{M}$.

Geometric setup in arbitrary ambient dimension. This geometric setup is easily extended for general m. The curvature vectors $\boldsymbol{n}_{t_0}$ and $\boldsymbol{b}_{t_0}$ are replaced with the principal directions, i.e., (unit) vectors indicating the $m-1$ normal directions of principal curvature. We can write these as $\boldsymbol{n}_{t_0,2}, \ldots, \boldsymbol{n}_{t_0,m}$. Together with $\boldsymbol{t}_{t_0}$, these vectors are mutually orthogonal. We set R to be the rotation matrix that maps $\boldsymbol{n}_{t_0,2}, \ldots, \boldsymbol{n}_{t_0,m}$ to $\boldsymbol{e}_2, \ldots, \boldsymbol{e}_m$, respectively, and we once again set $\tilde{\boldsymbol{f}} = R(\boldsymbol{f} - \boldsymbol{x}_0)$. Note that the principal directions and R should be chosen so that $R(\boldsymbol{t}_{t_0}) = \boldsymbol{e}_1$ (instead of $-\boldsymbol{e}_1$), for consistency.

The function $\tilde{\boldsymbol{f}} : \mathbb{R} \to \mathbb{R}^m$ can be described locally by $\tilde{\boldsymbol{f}}(x) = [x, \tilde{f}_2(x), \ldots, \tilde{f}_m(x)]^\mathsf{T}$, with

$$\begin{aligned}\tilde{f}_i(x) &= \frac{1}{2}\tilde{f}_i''(x)\, x^2 + \text{ higher-order terms} \\ &= \frac{1}{2}\kappa^{(i)} x^2 + \text{ higher-order terms},\end{aligned}$$

where $\kappa^{(i)}$ is the curvature of $\tilde{\boldsymbol{f}}$ at 0 (or equivalently, $\boldsymbol{f}$ at t_0) in the ith normal direction, $i = 2 \ldots, m$.

Leveraging the geometric setup to prove Lemma 6.4.1. Recall that our goal is to bound the distance between $\boldsymbol{z} := \boldsymbol{x}_0 + c\boldsymbol{u}$ and $\mathcal{M}$. We assume that $\|\boldsymbol{u}\|_2 = 1$ so that $\boldsymbol{u} = \pm \boldsymbol{t}_{t_0}$. The point corresponding to $\boldsymbol{z}$ in the rotated and shifted coordinate system is $[x_{\boldsymbol{z}}, 0, \ldots, 0]$, where $x_{\boldsymbol{z}} = c\,\langle \boldsymbol{u}, \boldsymbol{t}_{t_0} \rangle = c\,\mathrm{sgn}(\langle \boldsymbol{u}, \boldsymbol{t}_{t_0} \rangle)$.

By the geometric setup, we have that

$$\mathrm{dist}(\boldsymbol{z} - \mathcal{M}) = \min_{t \in \mathcal{D}} \|\boldsymbol{z} - \boldsymbol{f}(t)\|_2 \leq \min_{x \in [-\tilde{\epsilon}, \tilde{\epsilon}]} \|[x_{\boldsymbol{z}}, 0, \ldots, 0]^\mathsf{T} - \tilde{\boldsymbol{f}}(x)\|_2.$$

This, along with the observation that $\min_{x \in [-\tilde{\epsilon}, \tilde{\epsilon}]} \|[x_{\boldsymbol{z}}, 0, \ldots, 0]^\mathsf{T} - \tilde{\boldsymbol{f}}(x)\|_2 \leq \|[x_{\boldsymbol{z}}, 0, \ldots, 0] - \tilde{\boldsymbol{f}}(x_{\boldsymbol{z}})\|_2$, yields

$$\begin{aligned}\mathrm{dist}(\boldsymbol{z} - \mathcal{M}) &\leq \left\| [x_{\boldsymbol{z}}, 0, \ldots, 0]^\mathsf{T} - \tilde{\boldsymbol{f}}(x_{\boldsymbol{z}}) \right\|_2 \\ &= \left\| [x_{\boldsymbol{z}}, 0, \ldots, 0]^\mathsf{T} - [x_{\boldsymbol{z}}, \tilde{f}_2(x_{\boldsymbol{z}}), \ldots, \tilde{f}_m(x_{\boldsymbol{z}})]^\mathsf{T} \right\|_2\end{aligned}$$

$$
\begin{aligned}
&= \left\| [\tilde{f}_2(x_{\boldsymbol{z}}), \ldots, \tilde{f}_m(x_{\boldsymbol{z}})]^{\mathsf{T}} \right\|_2 \\
&\approx \left\| \left[\frac{1}{2}\kappa^{(2)} x_{\boldsymbol{z}}^2, \ldots, \frac{1}{2}\kappa^{(m)} x_{\boldsymbol{z}}^2 \right]^{\mathsf{T}} \right\|_2 \\
&= \frac{x_{\boldsymbol{z}}^2}{2} \Big(\sum_{i=2}^{m} (\kappa^{(i)})^2 \Big)^{1/2} \\
&= \frac{c^2}{2} \Big(\sum_{i=2}^{m} (\kappa^{(i)})^2 \Big)^{1/2},
\end{aligned}
$$

where the approximation follows from Taylor's expansion of the functions $\tilde{f}_i$, $i = 2, \ldots, m$, around the origin. It is clear from this sequence of equations that the distance between a point on the tangent line at $\boldsymbol{x}_0$ and the manifold $\mathcal{M}$ depends primarily on the scaling factor c (how close it is to $\boldsymbol{x}_0$ on the tangent line) and the principal curvatures in the $m-1$ normal directions.

Define the interval

$$
I_{x_{\boldsymbol{z}}} := \begin{cases} (0, x_{\boldsymbol{z}}), & x_{\boldsymbol{z}} \geq 0, \\ (x_{\boldsymbol{z}}, 0), & x_{\boldsymbol{z}} < 0. \end{cases}
$$

If we further assume that the third derivative $\tilde{f}_i'''(x)$ exists for all $x \in I_{x_{\boldsymbol{z}}}$ and that the second derivative $\tilde{f}_i''(x)$ is continuous for all $x \in \overline{I}_{x_{\boldsymbol{z}}}$, $i = 2, \ldots, m$, then Taylor's theorem states that

$$
\begin{aligned}
\|\boldsymbol{z} - \mathcal{M}\|_2 &\leq \left\| [\tilde{f}_2(x_{\boldsymbol{z}}), \ldots, \tilde{f}_m(x_{\boldsymbol{z}})]^{\mathsf{T}} \right\|_2 \\
&= \left\| \left[\frac{1}{2}\kappa^{(2)} x_{\boldsymbol{z}}^2 + \frac{\tilde{f}_2'''(\xi_2)}{3!} x_{\boldsymbol{z}}^3, \ldots, \frac{1}{2}\kappa^{(m)} x_{\boldsymbol{z}}^2 + \frac{\tilde{f}_m'''(\xi_m)}{3!} x_{\boldsymbol{z}}^3 \right]^{\mathsf{T}} \right\|_2 \\
&= \left(\sum_{i=2}^{m} \Big(\frac{1}{2}\kappa^{(i)} x_{\boldsymbol{z}}^2 + \frac{\tilde{f}_i'''(\xi_i)}{3!} x_{\boldsymbol{z}}^3 \Big)^2 \right)^{1/2} \\
&= \left(\sum_{i=2}^{m} \Big(\frac{1}{4}(\kappa^{(i)})^2 x_{\boldsymbol{z}}^4 + \frac{1}{6}\kappa^{(i)} \tilde{f}_i'''(\xi_i) x_{\boldsymbol{z}}^5 + \frac{1}{36} \Big(\tilde{f}_i'''(\xi_i) \Big)^2 x_{\boldsymbol{z}}^6 \Big) \right)^{1/2} \\
&= \left(\frac{x_{\boldsymbol{z}}^4}{4} \sum_{i=2}^{m} (\kappa^{(i)})^2 + \frac{x_{\boldsymbol{z}}^5}{6} \sum_{i=2}^{m} \tilde{f}_i'''(\xi_i) \Big(\kappa^{(i)} + \frac{x_{\boldsymbol{z}}}{6} \tilde{f}_i'''(\xi_i) \Big) \right)^{1/2} \\
&= \left(\frac{c^4}{4} \sum_{i=2}^{m} (\kappa^{(i)})^2 + \frac{x_{\boldsymbol{z}}^5}{6} \sum_{i=2}^{m} \tilde{f}_i'''(\xi_i) \Big(\kappa^{(i)} + \frac{x_{\boldsymbol{z}}}{6} \tilde{f}_i'''(\xi_i) \Big) \right)^{1/2},
\end{aligned}
$$

where each ξ_i is some number in $I_{x_{\boldsymbol{z}}}$, $i = 2, \ldots, m$. For simplicity, we write

$$\left\|[\tilde{f}_2(x_{\boldsymbol{z}}), \ldots, \tilde{f}_m(x_{\boldsymbol{z}})]^{\mathsf{T}}\right\|_2^2 = \frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R},$$

where

$$\mathcal{R} := \frac{x_{\boldsymbol{z}}^5}{6}\sum_{i=2}^{m}\tilde{f}_i'''(\xi_i)\Big(\kappa^{(i)} + \frac{x_{\boldsymbol{z}}}{6}\tilde{f}_i'''(\xi_i)\Big).$$

We want to bound $|\mathcal{R}|$. Since $I_{x_{\boldsymbol{z}}} \subset [-\tilde{\epsilon}, \tilde{\epsilon}]$ when $c \leq \tilde{\epsilon}$, then for any $x \in I_{x_{\boldsymbol{z}}}$, there exists $t \in I_\epsilon(\boldsymbol{x}_0)$ such that $x = \gamma(t) = \pm(t - t_0)$. Because $\boldsymbol{f}(t) = [f_1(t), \ldots, f_m(t)]^{\mathsf{T}}$ and $\tilde{\boldsymbol{f}}(x) = R(\boldsymbol{f}(t) - \boldsymbol{x}_0)$, then if R_{ij} denotes the (i,j)th entry of the rotation matrix R, it follows that

$$\begin{aligned}
|\tilde{f}_i'''(x)| &= |R_{i1}f_1'''(t) + \ldots + R_{im}f_m'''(t)| \\
&= |\left\langle [R_{i1}, \ldots, R_{im}]^{\mathsf{T}}, \boldsymbol{f}'''(t)\right\rangle| \\
&\leq \|\boldsymbol{f}'''(t)\|_2
\end{aligned}$$

by Cauchy-Schwarz, since R is an orthogonal matrix, $\gamma'(t) = \pm 1$, and $\gamma''(t) = \gamma'''(t) = 0$. So if we set

$$M := \max_{t \in I_\epsilon(\boldsymbol{x}_0)} \|\boldsymbol{f}'''(t)\|_2 = \max_{\boldsymbol{f}(t) \in \mathcal{N}_\epsilon(\boldsymbol{x}_0)} \|\boldsymbol{f}'''(t)\|_2 \quad \text{and} \quad \kappa := \max_{i \in \{2,\ldots,m\}} |\kappa^{(i)}|,$$

it follows that

$$\begin{aligned}
|\mathcal{R}| &= \left|\frac{x_{\boldsymbol{z}}^5}{6}\sum_{i=2}^{m}\tilde{f}_i'''(\xi_i)\Big(\kappa^{(i)} + \frac{x_{\boldsymbol{z}}}{6}\tilde{f}_i'''(\xi_i)\Big)\right| \\
&= \left|\frac{x_{\boldsymbol{z}}^5}{6}\right|\left|\sum_{i=2}^{m}\tilde{f}_i'''(\xi_i)\Big(\kappa^{(i)} + \frac{x_{\boldsymbol{z}}}{6}\tilde{f}_i'''(\xi_i)\Big)\right| \\
&\leq \left|\frac{(c\left\langle \boldsymbol{u}, \boldsymbol{t}_{t_0}\right\rangle)^5}{6}\right|\left[\sum_{i=2}^{m}\Big|\tilde{f}_i'''(\xi_i)\Big|\Big(|\kappa^{(i)}| + \Big|\frac{c\left\langle \boldsymbol{u}, \boldsymbol{t}_{t_0}\right\rangle}{6}\Big|\Big|\tilde{f}_i'''(\xi_i)\Big|\Big)\right] \\
&\leq \frac{c^5}{6}\left[\sum_{i=2}^{m}\Big|\tilde{f}_i'''(\xi_i)\Big|\Big(|\kappa^{(i)}| + \frac{c}{6}\Big|\tilde{f}_i'''(\xi_i)\Big|\Big)\right] \\
&\leq \frac{c^5}{6}\left[\sum_{i=2}^{m}M\Big(|\kappa^{(i)}| + \frac{c}{6}M\Big)\right]
\end{aligned}$$

$$\leq \frac{Mc^5}{6}(m-1)\Big(\kappa + \frac{Mc}{6}\Big).$$

This is sufficient to prove Lemma 6.4.1. The differentiability and continuity assumptions on $\boldsymbol{f}$ ensure that $\tilde{\boldsymbol{f}}$ is three times differentiable on $(-c, c)$ with continuous second-order derivatives on $[-c, c]$, so we can apply Taylor's theorem. □

6.4.2. Second Lemma. Again, recall that in LPCA-SRC we use the version of local PCA by Singer and Wu [**87**]. Thus Kaslovsky and Meyer's result [**57**] must be modified to hold in this case. We do this in Appendix A, and we denote the modified bound using Eq. (6.1).

LEMMA 6.4.2. *Let $\boldsymbol{x}_0$ be a point on the 1-dimensional manifold $\mathcal{M}$. Define $\boldsymbol{u}$ to be a unit basis vector of the tangent line at $\boldsymbol{x}_0$ on $\mathcal{M}$, and let $\hat{\boldsymbol{u}}$ be the approximation of $\boldsymbol{u}$ as found by Singer and Wu's local PCA [**87**]. Suppose that all the conditions of Kaslovsky and Meyer's modified tangent bound [**57**] hold so that $\|P - \widehat{P}\|_F \leq E^*$ with probability greater than $p := 1 - 2e^{-\xi_\lambda^2} - 9e^{-\xi^2}$. Then*

$$\|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2 \leq \sqrt{\frac{E^*}{2}}$$

with probability greater than p, where $P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})$ is the orthogonal projection of $\hat{\boldsymbol{u}}$ onto the subspace spanned by $\boldsymbol{u}$ (i.e., onto the tangent line at $\boldsymbol{x}_0$).

Proof of Lemma 6.4.2. Since $d = 1$, $\|P - \widehat{P}\|_F = \|\boldsymbol{u}\boldsymbol{u}^\mathsf{T} - \hat{\boldsymbol{u}}\hat{\boldsymbol{u}}^\mathsf{T}\|_F$. We can assume without loss of generality that $\langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \geq 0$, since $\boldsymbol{u}\boldsymbol{u}^\mathsf{T} = (-\boldsymbol{u})(-\boldsymbol{u})^\mathsf{T}$. Let the (i,j)th entry of $P := \boldsymbol{u}\boldsymbol{u}^\mathsf{T} \in \mathbb{R}^{m \times m}$ be given by $P_{ij} = u_i u_j$, where u_i denotes the ith coordinate of $\boldsymbol{u}$, $1 \leq i \leq m$. It follows that the Frobenius norm of the difference between P and $\widehat{P}$ satisfies

$$\begin{aligned}
\|P - \widehat{P}\|_F^2 &= \sum_{i=1}^{m}\sum_{j=1}^{m}(P_{ij} - \widehat{P}_{ij})^2 \\
&= \sum_{i=1}^{m}\sum_{j=1}^{m}(u_i u_j - \hat{u}_i \hat{u}_j)^2 \\
&= \sum_{i=1}^{m} u_i^2 \sum_{j=1}^{m} u_j^2 + \sum_{i=1}^{m} \hat{u}_i^2 \sum_{j=1}^{m} \hat{u}_j^2 - 2\sum_{i=1}^{m} u_i \hat{u}_i \sum_{j=1}^{m} u_j \hat{u}_j \\
&= 2 - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle^2 .
\end{aligned}$$

The modified Kaslovsky and Meyer bound [**57**] states that $\|P - \widehat{P}\|_F^2 = 2 - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \leq E^*$.

Case 1: $E^* > 2$. Of course, $2 - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \leq 2$. So if $E^* > 2$, the modified Kaslovsky and Meyer bound [**57**] is superfluous. In this case, it suffices to show that $\|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 \leq 1$, since $1 < \frac{E^*}{2}$. We have

$$\begin{aligned}\|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 &= \|\hat{\boldsymbol{u}} - \langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle \boldsymbol{u}\|_2^2 \\ &= \langle \hat{\boldsymbol{u}}, \hat{\boldsymbol{u}}\rangle - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 + \langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \langle \boldsymbol{u}, \boldsymbol{u}\rangle \\ &= 1 - \langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \leq 1. \ \square\end{aligned}$$

Case 2: $E^* \leq 2$. In this case, $1 - \frac{E^*}{2} \geq 0$, and we have

$$\begin{aligned}\|P - \widehat{P}\|_F^2 = 2 - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \leq E^* &\Rightarrow 1 - \frac{E^*}{2} \leq \langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle^2 \\ &\Rightarrow \sqrt{1 - \frac{E^*}{2}} \leq \langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle \\ &\Rightarrow \sqrt{1 - \frac{E^*}{2}} \leq \cos(\theta),\end{aligned}$$

where θ is the angle between $\boldsymbol{u}$ and $\hat{\boldsymbol{u}}$. The second-to-last inequality follows from $\langle \boldsymbol{u}, \hat{\boldsymbol{u}}\rangle \geq 0$.

Consider the right triangle with angle θ placed at the origin and incident sides $\hat{\boldsymbol{u}}$ and $P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})$. We want to bound the length of the third side, which is given by $\|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2$. Since $\|\hat{\boldsymbol{u}}\|_2 = 1$, the above inequality and basic trigonometry implies

$$\sqrt{1 - \frac{E^*}{2}} \leq \cos(\theta) = \|P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2. \tag{6.3}$$

By the Pythagorean theorem,

$$\|P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 + \|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 = 1. \tag{6.4}$$

Combining Eq. (6.3) and Eq. (6.4) produces

$$\begin{aligned}1 - \frac{E^*}{2} + \|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 \leq 1 \\ \Rightarrow \|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 \leq 1 - \left(1 - \frac{E^*}{2}\right)\end{aligned}$$

$$\Rightarrow \|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{u}})\|_2^2 \leq \frac{E^*}{2}. \ \square$$

COROLLARY 6.4.1. *Under the same hypotheses as Lemma 6.4.2, if we replace $\hat{\boldsymbol{u}}$ with its scaled version $\hat{\boldsymbol{v}} := c\hat{\boldsymbol{u}}$ for $c \geq 0$, then*

$$\|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 \leq \frac{c^2 E^*}{2}$$

with probability greater than p.

Proof of Corollary 6.4.1. We again assume without loss of generality that $\langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle > 0$ so that $\langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle \geq 0$.

Case 1: $E^* > 2$. As we saw above, the modified Kaslovsky and Meyer bound [**57**] is unhelpful, since

$$\|P - \widehat{P}\|_F^2 = 2 - 2\langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle^2 \leq 2.$$

The corollary still holds, however, as

$$\begin{aligned}
\|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 = \|\hat{\boldsymbol{v}} - \langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle \boldsymbol{u}\|_2^2 &= \langle \hat{\boldsymbol{v}}, \hat{\boldsymbol{v}} \rangle - 2\langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle^2 + \langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle^2 \langle \boldsymbol{u}, \boldsymbol{u} \rangle \\
&= \langle c\hat{\boldsymbol{u}}, c\hat{\boldsymbol{u}} \rangle - \langle \boldsymbol{u}, c\hat{\boldsymbol{u}} \rangle^2 \\
&= c^2 - c^2 \langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle^2 \\
&\leq c^2 < \frac{c^2 E^*}{2}. \ \square
\end{aligned}$$

Case 2: $E^* \leq 2$. In this case, $1 - \frac{E^*}{2} \geq 0$, and as we saw before, $\sqrt{1 - \frac{E^*}{2}} \leq \cos(\theta)$. We previously defined θ to be the angle between $\boldsymbol{u}$ and $\hat{\boldsymbol{u}}$, but of course, it is also the angle between $\boldsymbol{u}$ and $\hat{\boldsymbol{v}} = c\hat{\boldsymbol{u}}$. Similarly to above, we consider the right triangle formed by the vectors $\hat{\boldsymbol{v}}$ and $P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})$. We want to bound the length of the third side, $\|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2$. Since $\|\hat{\boldsymbol{v}}\|_2 = c$, it follows that

$$\begin{aligned}
\sqrt{1 - \frac{E^*}{2}} \leq \cos(\theta) = \frac{\|P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2}{c} \\
\Rightarrow c^2\Big(1 - \frac{E^*}{2}\Big) \leq \|P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2.
\end{aligned}$$

By the Pythagorean theorem,

$$\|P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 + \|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 = c^2,$$

and combining these equations produces

$$\begin{aligned} c^2\Big(1 - \frac{E^*}{2}\Big) + \|\hat{\boldsymbol{u}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 &\leq c^2 \\ \Rightarrow \|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2^2 \leq c^2 - c^2\Big(1 - \frac{E^*}{2}\Big) &\leq \frac{c^2 E^*}{2}. \quad \square \end{aligned}$$

6.4.3. Combining the Lemmas to Prove the Main Part of Theorem 6.3.1. We first review the notation and make a new definition. We have that $\boldsymbol{u}$ is a unit basis vector of the tangent line at $\boldsymbol{x}_0$ on $\mathcal{M}$ and $\hat{\boldsymbol{u}}$ is the approximation of $\boldsymbol{u}$ as found by Singer and Wu's local PCA [**87**]. Without loss of generality, we assume that $\langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \geq 0$. For scalar $c \geq 0$, we set $\hat{\boldsymbol{v}} := c\hat{\boldsymbol{u}}$. We can write

$$P_{\boldsymbol{u}}(\hat{\boldsymbol{v}}) = c \langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \boldsymbol{u} = \tilde{c}\boldsymbol{u},$$

where $\tilde{c} := c \langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \geq 0$. Set $\boldsymbol{v} := \tilde{c}\boldsymbol{u}$. Further, (re)define $\boldsymbol{z} := \boldsymbol{x}_0 + \boldsymbol{v}$ and $\hat{\boldsymbol{z}} := \boldsymbol{x}_0 + \hat{\boldsymbol{v}}$.

To prove the main part of Theorem 6.3.1, let $\mathcal{A}$ be the affine subspace given by $\mathcal{A} := \{\boldsymbol{x}_0 + \alpha\boldsymbol{u} : \alpha \in \mathbb{R}\}$. Then

$$\operatorname{dist}(\hat{\boldsymbol{z}} - \mathcal{M}) \leq \|\hat{\boldsymbol{z}} - P_{\mathcal{A}}(\hat{\boldsymbol{z}})\|_2 + \operatorname{dist}(P_{\mathcal{A}}(\hat{\boldsymbol{z}}) - \mathcal{M}). \tag{6.5}$$

Here, $P_{\mathcal{A}}(\hat{\boldsymbol{z}})$ is the orthogonal projection of $\hat{\boldsymbol{z}}$ onto $\mathcal{A}$, which can be written as $P_{\mathcal{A}}(\hat{\boldsymbol{z}}) = \boldsymbol{x}_0 + P_{\boldsymbol{u}}(\hat{\boldsymbol{z}} - \boldsymbol{x}_0)$. Observe that

$$\begin{aligned} \|\hat{\boldsymbol{z}} - P_{\mathcal{A}}(\hat{\boldsymbol{z}})\|_2 &= \|\hat{\boldsymbol{z}} - (\boldsymbol{x}_0 + P_{\boldsymbol{u}}(\hat{\boldsymbol{z}} - \boldsymbol{x}_0))\|_2 \\ &= \|(\boldsymbol{x}_0 + \hat{\boldsymbol{v}}) - \boldsymbol{x}_0 - P_{\boldsymbol{u}}(\hat{\boldsymbol{z}} - \boldsymbol{x}_0)\|_2 \\ &= \|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}((\boldsymbol{x}_0 + \hat{\boldsymbol{v}}) - \boldsymbol{x}_0)\|_2 \\ &= \|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2, \end{aligned}$$

and

$$\begin{aligned}\operatorname{dist}(P_{\mathcal{A}}(\hat{\boldsymbol{z}}) - \mathcal{M}) &= \operatorname{dist}((\boldsymbol{x}_0 + P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})) - \mathcal{M}) \\ &= \operatorname{dist}((\boldsymbol{x}_0 + \langle \boldsymbol{u}, \hat{\boldsymbol{v}} \rangle \boldsymbol{u}) - \mathcal{M}) \\ &= \operatorname{dist}((\boldsymbol{x}_0 + c_1 \langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \boldsymbol{u}) - \mathcal{M}) \\ &= \operatorname{dist}((\boldsymbol{x}_0 + \boldsymbol{v}) - \mathcal{M}) \\ &= \operatorname{dist}(\boldsymbol{z} - \mathcal{M}).\end{aligned}$$

Plugging these findings into Eq. (6.5) renders

$$\operatorname{dist}(\hat{\boldsymbol{z}} - \mathcal{M}) \leq \|\hat{\boldsymbol{v}} - P_{\boldsymbol{u}}(\hat{\boldsymbol{v}})\|_2 + \operatorname{dist}(\boldsymbol{z} - \mathcal{M}). \tag{6.6}$$

The proof of the main part of Theorem 6.3.1 follows by applying Corollary 6.4.1 and Lemma 6.4.1 to the first and second terms of the right side of Eq. (6.6), respectively, and noting that $\tilde{c} = c \langle \boldsymbol{u}, \hat{\boldsymbol{u}} \rangle \leq c$. $\square$

6.4.4. Completing the Proof. In practice, we may not be able to compute the exact value of the bound δ in Eq. (6.2), since $\mathcal{R}$ depends on the third derivative of $\boldsymbol{f}$. Instead, we define an approximation of this value and use R_{bound} (an upper bound on $|\mathcal{R}|$) to bound the difference between our approximation and δ.

Let

$$\delta_{\text{approx}} := c\sqrt{\frac{E^*}{2}} + \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2}.$$

This is the result of setting $\mathcal{R} = 0$ in Eq. (6.2). We have

$$\begin{aligned}|\delta - \delta_{\text{approx}}| &= \left|\left[c\sqrt{\frac{E^*}{2}} + \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2}\right] - \left[c\sqrt{\frac{E^*}{2}} + \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2}\right]\right| \\ &= \left|\left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2}\right|\end{aligned}$$

$$= \begin{cases} \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2} & \text{if } \mathcal{R} \geq 0 \\ \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2} & \text{if } \mathcal{R} < 0. \end{cases}$$

Further, since we know that $|\mathcal{R}| \leq R_{\text{bound}}$, we have that

$$\left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 - R_{\text{bound}}\right)^{1/2} \leq \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + \mathcal{R}\right)^{1/2} \leq \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + R_{\text{bound}}\right)^{1/2},$$

which produces

$$|\delta - \delta_{\text{approx}}| \leq \begin{cases} \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + R_{\text{bound}}\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2} & \text{if } \mathcal{R} \geq 0 \\ \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 - R_{\text{bound}}\right)^{1/2} & \text{if } \mathcal{R} < 0. \end{cases}$$

Thus setting

$$\zeta_{\text{bound}} := \max\left\{ \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 + R_{\text{bound}}\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2}, \right.$$
$$\left. \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2\right)^{1/2} - \left(\frac{c^4}{4}\sum_{i=2}^{m}(\kappa^{(i)})^2 - R_{\text{bound}}\right)^{1/2} \right\}$$

implies that $|\delta - \delta_{\text{approx}}| \leq \zeta_{\text{bound}}$. This completes the proof of Theorem 6.3.1. □

6.5. Remarks Regarding Implementation

Though Theorem 6.3.1, in particular Lemma 6.4.1, heavily involves a description of the manifold $\mathcal{M}$ via a function $\boldsymbol{f}$, we stress that $\boldsymbol{f}$ does not need to be found explicitly. The properties of $\boldsymbol{f}$ stated in Theorem 6.3.1 can be estimated and/or ascertained directly from $\mathcal{M}$. In particular, the principal curvatures can be estimated from points on the manifold, and the required continuity and differentiability conditions of $\boldsymbol{f}$ at $\boldsymbol{x}_0$ depend directly on the smoothness of $\mathcal{M}$.

It is important to note that computing the modified Kaslovsky and Meyer tangent bound in Eq. (6.1) may be somewhat difficult in practice due to ascertaining the values of n, r, K, σ, ξ, and ξ_λ. However, the authors give some good suggestions for determining these parameters. For

example, see their paper [**57**] for references for estimating the curvatures contained in K and the noise σ. The radius parameter r, which denotes the maximum distance between the local origin and the neighbors in local PCA *as measured along the tangent plane*, cannot be computed in practice (because the true tangent plane is unknown). However, Kaslovsky and Meyer [**57**] suggest that this parameter be estimated using

$$\hat{r}(s) := \sqrt{\frac{1}{2\gamma}(-1 + \sqrt{4\gamma(s - \sigma^2 m)})} \approx r,$$

where s bounds the squared distance between the neighboring samples and the local origin (e.g., $s := \|\boldsymbol{x}_0 - \boldsymbol{x}_{n+1}\|_2^2$ using Algorithm 4 in LPCA-SRC) and

$$\gamma := \frac{1}{10}\|\boldsymbol{\kappa}\|_2^2$$

in the case $d = 1$. Additionally, the probability constants ξ and ξ_λ should be chosen to satisfy the constraints of Kaslovsky and Meyer's (modified) tangent bound and so that the result holds with sufficiently high probability. Lastly, the tangent bound itself provides a means of estimating the number of samples/neighbors parameter n: Kaslovsky and Meyer state that n should be chosen to (empirically) minimize the tangent bound [**57**]. Thus, under generous sampling conditions, doing so could potentially be an alternative to setting n using cross-validation in LPCA-SRC.

To address the issue of obtaining a clean local origin $\boldsymbol{x}_0$, Kaslovsky and Meyer proposed an efficient algorithm that "tracks" the center of mass of the neighbors of a noisy sample over increasingly large neighborhood radii. The geometric information obtained during this procedure is used to compute a point $\hat{\boldsymbol{x}}_0 \approx \boldsymbol{x}_0$. See Section 5.1 of their paper [**57**] for more details. In the context of LPCA-SRC, this algorithm could be used to estimate a clean point from each training sample. The tangent vectors would then be centered at these points instead of the original training samples. We expect that this method would also improve the classification accuracy of LPCA-SRC, though it would increase its computational expense.

Despite these proposed solutions, there is still reason to doubt the practical functionality of Theorem 6.3.1. The information we require from $\mathcal{M}$ may be impractical to obtain in real-world classification problems for which the true class manifolds are unknown. Though the conditions of Lemma 6.4.1 are satisfied for sufficiently smooth $\mathcal{M}$ and sufficiently small c, the precise conditions

may be difficult to check when our knowledge of $\mathcal{M}$ is based entirely on a limited number of sampled points. Further, this problem is exacerbated when the manifold is sparsely-sampled. We admit that this is a substantial shortcoming in the usefulness of Theorem 6.3.1. As a related issue, Theorem 6.3.1 may not hold under the sparse sampling conditions common in classification problems such as face recognition, due to its reliance on Kaslovsky and Meyer's tangent bound, which requires generous sampling [**57**]. Another weakness, of course, is that Theorem 6.3.1 is only stated for the case $d = 1$. The above analysis is merely a first step towards quantifying the usefulness of the tangent vectors in LPCA-SRC, and more work is needed.

CHAPTER 7

Other Local PCA Modifications

In this chapter, we use local PCA to modify two other kinds of representation-based algorithms, analogous to our modification of SRC in LPCA-SRC as presented in Chapter 4. Namely, we consider CRC-RLS [**114**] (CRC with ℓ^2-regularization) and *classification via structured sparsity* [**41**] (a method that seeks a *block-sparse* representation of the test sample over the set of training samples). Note that block-sparsity is also referred to as *group-sparsity* in the literature. We compare the original and modified versions of these algorithms with SRC, LPCA-SRC, and $\text{SRC}_{\text{pruned}}$.

7.1. Modifying CRC-RLS

We discussed in Chapter 3 the computational efficiency of using ℓ^2-regularization (instead of ℓ^1-regularization) in the CRC framework. The key to this efficiency is that the matrix $P := (X_{\text{tr}}^{\mathsf{T}} X_{\text{tr}} + \lambda I)^{-1} X_{\text{tr}}^{\mathsf{T}}$ may be computed offline, and so for any number of test samples, solving Eq. (3.9) with $p = q = 2$ requires only the matrix-vector multiplication $\boldsymbol{\alpha}^* = P\boldsymbol{y}$.

Since the pruning of the dictionary in LPCA-SRC is computed online, we cannot obtain the same computational speedup in LPCA-SRC by simply replacing the ℓ^1-norm in Eq. (4.2) with the (squared) ℓ^2-norm. In fact, as our experimental results will demonstrate, the matrix multiplication required to compute $(D_{\boldsymbol{y}}^{\mathsf{T}} D_{\boldsymbol{y}} + \lambda I)^{-1} D_{\boldsymbol{y}}^{\mathsf{T}}$, even when QR-factorization is used, can be significantly slower than using HOMOTOPY to solve the original ℓ^1-minimization problem, especially when the dictionary is large. However, our primary focus in the following experiments will be on how this non-sparsifying method of regularization (i.e., ℓ^2-minimization) affects classification accuracy.

7.1.1. Methods Compared. We compare SRC, LPCA-SRC, $\text{SRC}_{\text{pruned}}$, and CRC-RLS with the following modified methods:

- *LPCA-CRC*: LPCA-SRC, but with Eq. (4.2) replaced with

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\boldsymbol{y}}}} \left\{ \|\boldsymbol{y} - D_{\boldsymbol{y}}\boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_2^2 \right\}, \tag{7.1}$$

- *CRC pruned*: $\text{SRC}_{\text{pruned}}$ with the analogous modification,
- *CRC unnorm.*: This modification of CRC solves

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\text{tr}}}} \left\{ \|\boldsymbol{y} - X_{\text{tr}}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_2^2 \right\} \tag{7.2}$$

and then classifies $\boldsymbol{y}$ according to

$$\text{class\_label}(\boldsymbol{y}) = \arg\min_{1\le l\le L} \left\| \boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}^*) \right\|_2. \tag{7.3}$$

Note that $\text{CRC}_{\text{unnorm.}}$ is SRC with the ℓ^1-norm replaced with the squared ℓ^2-norm (and without the coefficient of 1/2 on the approximation error). Alternatively, it is exactly CRC-RLS without the normalization factor in Eq. (3.10): "unnorm." abbreviates "unnormalized."

7.1.2. Experimental Results. We compared the above methods on the sinusoidal waveform synthetic database described in Section 5.3 (for a range of training class sizes and noise levels) and on AR-1, the version of the AR database that does not contain images with occlusion. The results were computed over 100 trials for the synthetic database and 10 trials for AR-1. The parameter λ was set to 0.001 in the original algorithms and set using cross-validation in the algorithms LPCA-CRC, CRC-RLS, $\text{CRC}_{\text{unnorm.}}$, and $\text{CRC}_{\text{pruned}}$.

Figure 7.1 shows the accuracy results on the synthetic database as the number of training samples in each class was varied. As we can see, none of the ℓ^2-based algorithms were competitive for small values of N_0, though both LPCA-CRC and $\text{CRC}_{\text{pruned}}$ improved significantly as N_0 increased. The dictionary pruning steps in these methods can be interpreted as an enforcement of (some degree of) sparsity, in the sense that the eliminated training samples are viewed as samples with coefficients equal to 0 in the approximation of $\boldsymbol{y}$ over the full dictionary before pruning. This additional aspect of sparsity (for all but $N_0 = 5$ at which no dictionary pruning occurred) gave LPCA-CRC and $\text{CRC}_{\text{pruned}}$ an advantage over the denser ℓ^2-based methods. Even at $N_0 = 75$, however, the accuracies of LPCA-CRC and $\text{CRC}_{\text{pruned}}$ were still exceeded by those of all three ℓ^1-based algorithms.

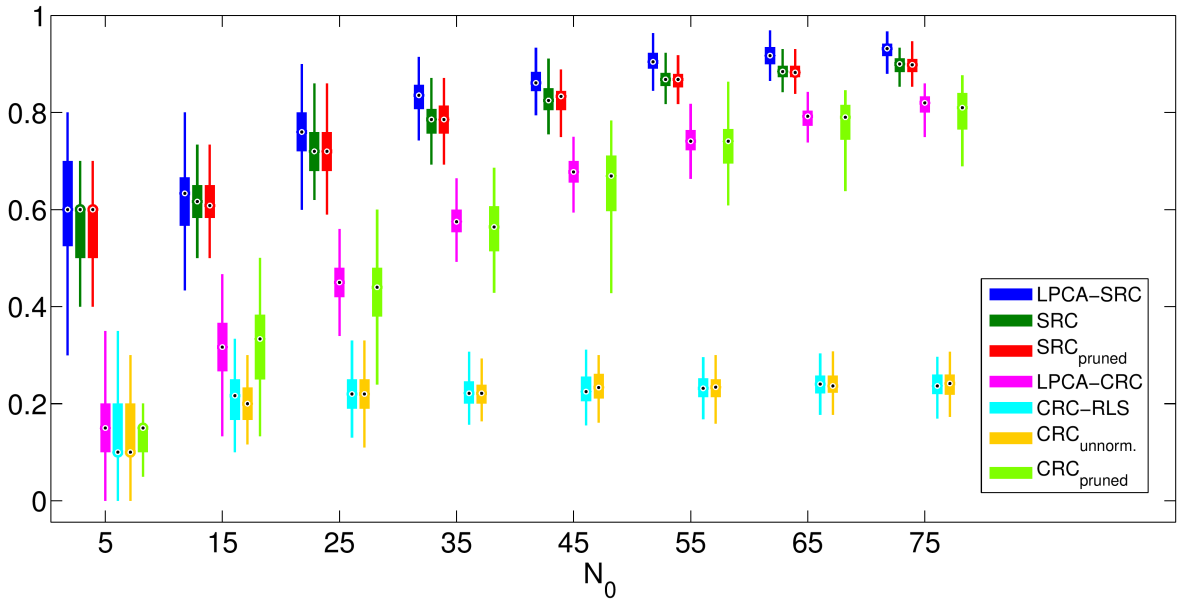


FIGURE 7.1. Box plot of the average classification accuracy (over 100 trials) of the original and ℓ^2-regularized versions of algorithms on the synthetic database with varying training class size N_0. We fixed $\eta = 0.001$.

Though it may be difficult to see in Figure 7.1, LPCA-CRC generally outperformed $\text{CRC}_{\text{pruned}}$, with the advantage of LPCA-CRC over $\text{CRC}_{\text{pruned}}$ decreasing slightly as N_0 increased, as in the case of the original algorithms LPCA-SRC and $\text{SRC}_{\text{pruned}}$. The tangent vectors in LPCA-CRC also helped to stabilize classification accuracy, as LPCA-CRC typically had lower variance than $\text{CRC}_{\text{pruned}}$ for sufficiently large N_0. Lastly, the nearly identical behavior of CRC-RLS and $\text{CRC}_{\text{unnorm.}}$ suggests that the normalization factor in the classification stage of CRC-RLS makes very little difference on this database, though as both methods had classification accuracy comparable to random guessing, this is far from conclusive.

Figure 7.2 shows the results as the noise level η was varied on the synthetic database. Again, though LPCA-CRC and $\text{CRC}_{\text{pruned}}$ performed significantly better than the ℓ^2-based algorithms without dictionary pruning, they were still clearly outperformed by the methods that use ℓ^1-minimization. Though their classification accuracy was only slightly affected by the increase in noise for the displayed values of η, additional experiments confirmed that for $\eta > 0.5$, LPCA-CRC and $\text{CRC}_{\text{pruned}}$ continue to be outperformed by the ℓ^1-based methods until all algorithms converge at the point that class structure is entirely lost. Again, the tangent vectors in LPCA-CRC lent it

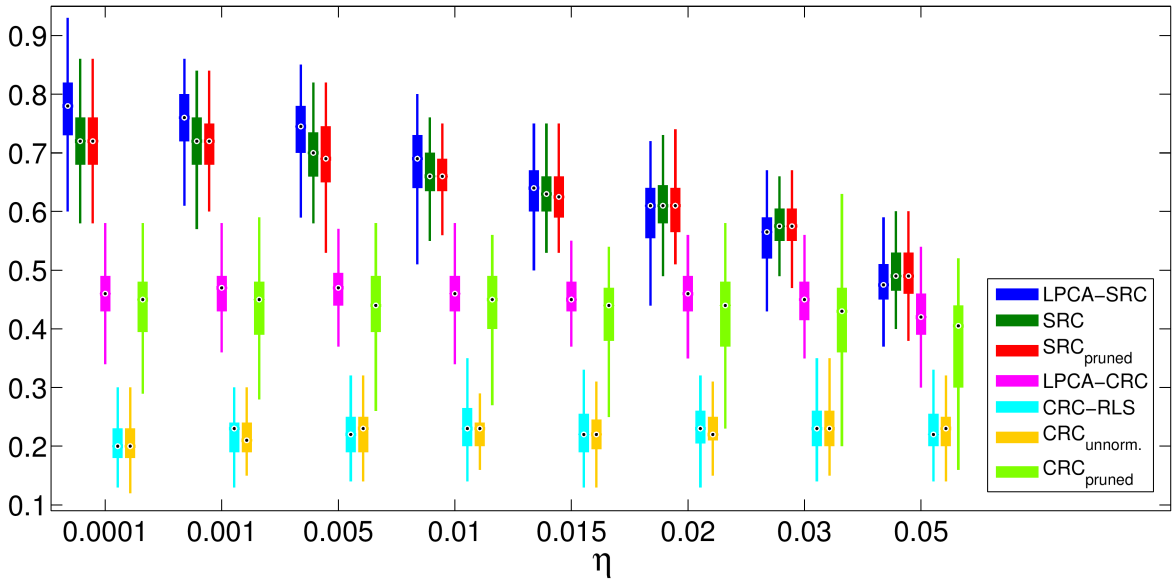


FIGURE 7.2. Box plot of the average classification accuracy (over 100 trials) of the original and ℓ^2-regularized versions of algorithms on the synthetic database with varying noise level η. We fixed $N_0 = 25$.

an advantage over $\text{CRC}_{\text{pruned}}$ (by an average of 3.5%), and the normalization factor in CRC-RLS had no discernible effect.

The runtime of each algorithm on the synthetic database is displayed in Table 7.1 for every other value of N_0 and η. As we can see, the ℓ^2-methods generally offered a significant speedup over the methods that use ℓ^1-regularization. The exception, however, was LPCA-CRC, which ran in about the same amount of time as LPCA-SRC. We also observe that the runtimes of the ℓ^1-methods became larger as η increased. This is the result of a small increase in both the number of iterations and the runtime of each iteration in the ℓ^1-minimization algorithm HOMOTOPY, due to the additional noise requiring (relatively) denser coefficient vectors. (The coefficient vectors in each ℓ^1-method had on average 2-3 nonzero coefficients when $\eta = 0.001$; this increased to about 5 nonzero coefficients when $\eta = 0.05$.) More details regarding the efficiency of HOMOTOPY as it relates to the sparsity of its solution can be found in the paper by Donoho and Tsaig [**33**]. Note that the closed-form solutions for the coefficient vectors used in the ℓ^2-regularized methods (once their dictionaries were constructed) prevented their runtimes from being similarly affected by the increase in noise.

	Varying N_0 ($\eta = 0.001$)				Varying η ($N_0 = 25$)			
Algorithm	$N_0 = 5$	$N_0 = 25$	$N_0 = 45$	$N_0 = 65$	$\eta = 0.001$	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.05$
LPCA-SRC	0.011	0.069	0.115	0.137	0.074	0.082	0.096	0.112
SRC	0.004	0.040	0.105	0.126	0.042	0.044	0.052	0.064
$\text{SRC}_{\text{pruned}}$	0.007	0.054	0.130	0.167	0.056	0.063	0.067	0.077
LPCA-CRC	0.015	0.079	0.100	0.139	0.072	0.101	0.074	0.100
CRC-RLS	0.001	0.005	0.010	0.017	0.005	0.005	0.005	0.005
$\text{CRC}_{\text{unnorm.}}$	0.001	0.004	0.009	0.015	0.004	0.004	0.004	0.004
$\text{CRC}_{\text{pruned}}$	0.006	0.045	0.064	0.085	0.042	0.038	0.040	0.047

TABLE 7.1. Average runtime (in seconds) of the original and ℓ^2-regularized versions of algorithms over 100 trials on the synthetic database with varying training class size N_0 and varying noise level η.

We now consider the performance of the various algorithms on the face database AR-1. Table 7.2 contains the average accuracy, standard error, and runtime over 10 trials. We notice that all the ℓ^2-based algorithms became competitive as the dimension increased, performing comparably to their ℓ^1-based counterparts for $m_{\text{PCA}} = 120$. Despite their fast runtimes, CRC-RLS and $\text{CRC}_{\text{unnorm.}}$ had significantly worse performance than the other methods for $m_{\text{PCA}} \in \{30, 56\}$; additionally, the normalization factor in CRC-RLS was, if anything, counterproductive. This is in contrast to the CRC authors' experimental finding that the normalization factor slightly improved classification accuracy [**114**]. It could be the case that this is true at higher feature dimensions or on other databases.[1] On the other hand, $\text{CRC}_{\text{pruned}}$ had comparable accuracy to SRC and $\text{SRC}_{\text{pruned}}$ at all feature dimensions, though it offered very little advantage in runtime. Lastly, as we mentioned at the beginning of this section, LPCA-CRC was *significantly* slower than LPCA-SRC for all tried values of m_{PCA}. Though it had accuracy comparable to or exceeding that of SRC and $\text{SRC}_{\text{pruned}}$ at all three feature dimensions, it was still outperformed by the original LPCA-SRC for $m_{\text{PCA}} \in \{30, 56\}$. Thus there appear to be only disadvantages—in terms of both classification accuracy and computational efficiency—in using LPCA-CRC instead of LPCA-SRC.

Lastly, we consider the difference in sparsity between the ℓ^1-minimized and ℓ^2-minimized coefficient vectors. Table 7.3 displays the average number of nontrivial coefficients (i.e., coefficients with magnitude at least 0.001) in the solution vector $\boldsymbol{\alpha}^*$ in the compared algorithms. We normalized

[1]The CRC authors ran experiments on AR-1 with $m_{\text{PCA}} \in \{54, 120\}$ (as well as $m_{\text{PCA}} = 300$); however, their images were cropped to a different size than we used, and they selected their training set differently than we did [**114**]. They also did not report the classification accuracy of CRC-RLS without the normalization factor, and so it is difficult to precisely explain our contrasting findings.

	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	Acc	SE	t	Acc	SE	t	Acc	SE	t
LPCA-SRC	**0.8663**	4.1	7.25	**0.9544**	2.3	12.50	0.9711	1.7	19.07
SRC	0.8273	4.2	6.11	0.9357	2.6	8.87	0.9631	1.6	13.57
$\text{SRC}_{\text{pruned}}$	0.8277	4.8	3.76	0.9353	3.8	5.10	0.9651	1.8	6.90
LPCA-CRC	0.8339	6.9	121.09	0.9409	1.5	385.81	**0.9729**	2.1	269.69
CRC-RLS	0.7516	4.9	0.64	0.9080	3.6	0.67	0.9611	2.0	0.74
$\text{CRC}_{\text{unnorm.}}$	0.7584	5.2	0.54	0.9070	3.1	0.58	0.9650	2.5	0.63
$\text{CRC}_{\text{pruned}}$	0.8357	4.4	5.51	0.9366	2.1	5.45	0.9659	2.1	5.57

TABLE 7.2. Average accuracy, standard error ($\times 10^{-3}$), and runtime (in seconds) of the original and ℓ^2-regularized versions of algorithms over 10 trials on AR-1.

this number by the length of the (respective) dictionary. Since the coefficient vectors are the same in CRC-RLS and $\text{CRC}_{\text{unnorm.}}$, we only show the results for the latter algorithm.

	Synthetic ($\eta = 0.001$)				Synthetic ($N_0 = 25$)				AR-1		
Algorithm	$N_0 = 5$	25	45	65	$\eta = 0.001$	0.01	0.02	0.05	$m_{\text{PCA}} = 30$	56	120
LPCA-SRC	0.15	0.12	0.21	0.19	0.13	0.06	0.08	0.09	0.22	0.53	0.65
LPCA-CRC	0.98	0.98	0.99	0.99	0.99	0.97	0.98	0.98	0.78	0.74	0.82
SRC	0.12	0.13	0.07	0.04	0.19	0.04	0.04	0.06	0.19	0.40	0.82
$\text{CRC}_{\text{unnorm.}}$	0.99	0.97	0.94	0.92	0.97	0.98	0.98	0.99	0.87	0.90	0.92
$\text{SRC}_{\text{pruned}}$	0.13	0.14	0.08	0.02	0.17	0.05	0.07	0.13	0.47	0.84	0.96
$\text{CRC}_{\text{pruned}}$	0.99	0.99	0.99	1.00	0.99	0.99	0.99	0.99	0.95	0.96	0.97

TABLE 7.3. Average number of nontrivial coefficients (divided by the total number of coefficients) in the original and ℓ^2-regularized versions of the algorithms on the synthetic database (100 trials) and AR-1 (10 trials).

It should come as no surprise that on all versions of the synthetic database, the ℓ^1-minimized coefficient vectors were *significantly* sparser than their ℓ^2-based counterparts. The results on AR-1, however, are intriguing: Though the coefficient vectors of the original methods LPCA-SRC, SRC, and $\text{SRC}_{\text{pruned}}$ were relatively sparse at $m_{\text{PCA}} = 30$ (for example, the methods LPCA-SRC and SRC used only a quarter of the nontrivial coefficients required by LPCA-CRC and $\text{CRC}_{\text{unnorm.}}$, respectively), they became denser as the feature dimension increased. At $m_{\text{PCA}} = 120$, for instance, $\text{SRC}_{\text{pruned}}$ and $\text{CRC}_{\text{pruned}}$ both had nearly maximally-dense coefficient vectors. The reason for this increasing number of nontrivial coefficients in the ℓ^1-based algorithms is likely due to the redundancy of the dictionary decreasing (i.e., m_{PCA} becoming closer to N). This helps to explain why the ℓ^2-regularized methods performed comparably to the original algorithms at $m_{\text{PCA}} = 120$.

REMARK 7.1.1. *We could have solved the optimization problem in all compared algorithms using Friedman et al.'s* `glmnet` *package [**43**, **86**], a coordinate-descent approach to solving generalized linear models*[2] *with either ℓ^1-regularization, ℓ^2-regularization, or a combination of both (this is called* elastic-net*). In the case of LPCA-CRC especially, this approach is much more efficient than the closed-form matrix multiplication discussed above. However, it is recommended that the columns of the input dictionary in* `glmnet` *be centered so that their means are 0.*[3] *Since the mean of each sample (generally) contains class information, this pre-processing significantly decreased the classification accuracy of LPCA-SRC, SRC, and $\text{SRC}_{\text{pruned}}$ compared to our experiments in Chapter 5 using HOMOTOPY.*

An additional deterrent in using `glmnet` *to solve the optimization problems in representation-based classification algorithms involves the specification of the regularization parameter λ. Designed primarily for regression applications,* `glmnet` *does not generally work well with values of λ too close to 0, as the resulting solution would generally correspond to overfitting. In the case that we want $\boldsymbol{y}$ to be a nearly-exact linear combination of the dictionary elements (i.e., $\lambda \approx 0$), proper use of* `glmnet` *will often return a solution corresponding to an undesirably large value of λ.*

7.2. Modifying Structured Sparsity/Block-Sparse Methods

In SRC, we assume that the test sample can be well-approximated by its same class training samples, with the goal being that nonzero coefficients in the ℓ^1-minimized approximation of $\boldsymbol{y}$ occur primarily at training samples in that class. However, as the desired coefficient vector has the form

$$\boldsymbol{\alpha} = [0, \ldots, 0, \alpha_1^{(l)}, \ldots, \alpha_{N_l}^{(l)}, 0, \ldots, 0]^{\mathsf{T}} \in \mathbb{R}^{N_{\text{tr}}}$$

for $\boldsymbol{y}$ in class l, seeking the sparsest vector is not necessarily the most direct approach, especially if the number of classes L is small. Instead, one could seek a representation of the test sample that constrains the number of *classes* that contain training samples with nonzero coefficients. In other words, given that the columns of X_{tr} are grouped by class so that we can write $X_{\text{tr}} = [X^{(1)}, \ldots, X^{(L)}]$, then if we partition $\boldsymbol{\alpha}$ into the "blocks" $\boldsymbol{\alpha} = [(\boldsymbol{\alpha}^{(1)})^{\mathsf{T}}, \ldots, (\boldsymbol{\alpha}^{(L)})^{\mathsf{T}}]^{\mathsf{T}}$ where $\boldsymbol{\alpha}^{(l)} =$

[2]The least squares terms in Eq. (3.7) and Eq. (7.2), for example, make these *linear* models.

[3]Centering is necessary in our application because it ensures that the *regression intercept* in the `glmnet` optimization problem is equal to 0, as is required for representation-based classification. See Hastie et al.'s book [**49**] for more details.

$[\alpha_1^{(l)}, \ldots, \alpha_{N_l}^{(l)}]^\mathsf{T} \in \mathbb{R}^{N_l}$, $1 \leq l \leq L$, then we could aim to minimize the number of blocks $\boldsymbol{\alpha}^{(l)}$ that contain nonzero coefficients. Of course, the ideal solution has nonzero coefficients in exactly one block, and it is assumed that this block corresponds to the correct class.

In one variant of the *structured sparse representation* (SSR) classification algorithm of Elhamifar and Vidal [**40**], the optimization problem behind this concept is written as

$$\boldsymbol{\alpha}^* := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{tr}}}} \sum_{l=1}^{L} \mathbb{I}(\|\boldsymbol{\alpha}^{(l)}\|_2 > 0) \text{ subject to } \boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}. \tag{7.4}$$

However, as in the case of finding the sparsest coefficient vector (via ℓ^0-"norm" minimization), solving Eq. (7.4) is intractable. Fortunately, the relaxation

$$\boldsymbol{\alpha}^* := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{tr}}}} \sum_{l=1}^{L} \|\boldsymbol{\alpha}^{(l)}\|_2 \text{ subject to } \boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha} \tag{7.5}$$

has been proven to recover the solution to Eq. (7.4) under certain conditions [**37**, **41**]. Note that the objective function is a special case of the so-called *mixed ℓ^p/ℓ^q-norm*:

$$\Big(\sum_{l=1}^{L} \|\boldsymbol{\alpha}^{(l)}\|_p^q \Big)^{1/q},$$

with $p = 2$ and $q = 1$. Much as minimizing the ℓ^1-norm can be used to determine the sparsest solution in certain situations, minimizing the mixed ℓ^2/ℓ^1-norm can be used to retrieve what we call the *block-sparsest solution*, i.e., the solution with nonzeros in the minimal number of blocks [**37**]. This is referred to as *block-sparse recovery* (BSR). Classification in SSR is then determined using the solution $\boldsymbol{\alpha}^*$ of Eq. (7.5) in Eq. (3.6), as in SRC.

When noise makes the exact representation $\boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}$ impractical, we can instead solve

$$\boldsymbol{\alpha}^* := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{tr}}}} \|\boldsymbol{y} - X_{\text{tr}}\boldsymbol{\alpha}\|_2^2 + \lambda \sum_{l=1}^{L} \|\boldsymbol{\alpha}^{(l)}\|_2. \tag{7.6}$$

In this format, BSR is easily identifiable as the *group-LASSO* (*Least Absolute Shrinkage and Selection Operator*), which was originally proposed by Yuan and Lin to solve regression problems [**112**].

On the other hand, Eq. (7.6) is equivalent to

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\text{tr}}}} \sum_{l=1}^{L} \|\boldsymbol{\alpha}^{(l)}\|_2 \text{ subject to } \|\boldsymbol{y} - X_{\text{tr}}\boldsymbol{\alpha}\|_2 \leq \epsilon, \tag{7.7}$$

the formulation used for robust block-sparse signal recovery by Eldar and Mishali [**37**].

7.2.1. Methods Compared. We compared SRC, LPCA-SRC, and $\text{SRC}_{\text{pruned}}$ with this version of SSR, formulating the optimization problem as in Eq. (7.7). For fair comparison, we computed new results for the first three algorithms with their respective ℓ^1-minimization problems formulated as

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N}} \|\boldsymbol{\alpha}\|_1 \text{ subject to } \|\boldsymbol{y} - D\boldsymbol{\alpha}\|_2 \leq \epsilon, \tag{7.8}$$

where $D \in \mathbb{R}^{m\times N}$ denotes the algorithm's dictionary. Note that this formulation is equivalent to each method's previously presented optimization problem under an appropriate correspondence between the parameters ϵ and λ; however, this correspondence cannot be determined a priori.

We also considered the relative classification performance of the modified algorithms LPCA-SSR, wherein Eq. (4.2) is replaced with

$$\boldsymbol{\alpha}^* := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^{N_{\text{tr}}}} \sum_{l=1}^{L} \|\boldsymbol{\alpha}^{(l)}\|_2 \text{ subject to } \|\boldsymbol{y} - D_{\boldsymbol{y}}\boldsymbol{\alpha}\|_2 \leq \epsilon, \tag{7.9}$$

and $\text{SSR}_{\text{pruned}}$, which is $\text{SRC}_{\text{pruned}}$ with a similar revision.

To solve the optimization problems in all six methods, we used the sparse reconstruction toolbox SPGL1 [**95**, **96**], due to its easily-implementable methods for solving both Eq. (7.7) and Eq. (7.8). The parameter ϵ was set using cross-validation in all classification algorithms.

7.2.2. Experimental Results. As in the previous section, we compared the above algorithms on the sinusoidal waveform synthetic database from Chapter 5 and the face database AR-1.

Figure 7.3 shows the results as the training class size N_0 was varied on the synthetic database. The immediate observation is that SSR performed significantly worse than SRC and the other ℓ^1-based methods. Though the theoretical motivation behind SSR and that of CRC-RLS discussed in the previous section are completely different, their reasons for failure on this database are essentially the same: since each training class can generally provide a decent approximation of the test sample,

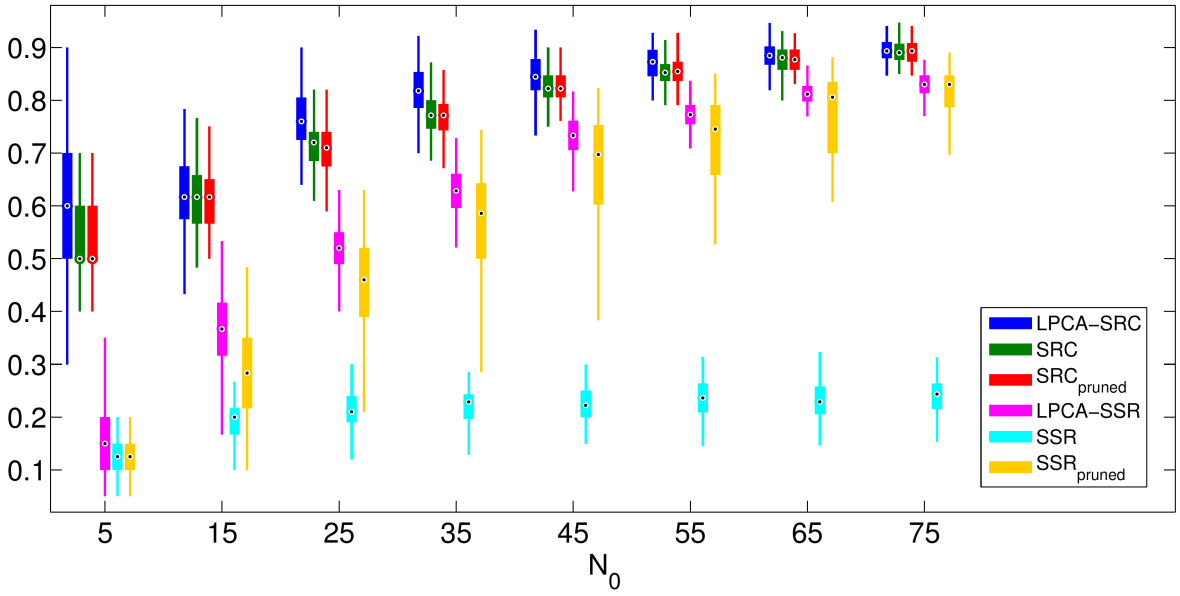


FIGURE 7.3. Box plot of the average classification accuracy (over 100 trials) of the original and SSR versions of algorithms on the synthetic database with varying training class size N_0. We fixed $\eta = 0.001$.

the classes must be differentiated *locally*, and these algorithms do not have the machinery to do this. The block-sparse prior behind SSR only considers each class as a whole, and so it cannot identify the class manifold structure in a small neighborhood around $\boldsymbol{y}$. As in the case of the ℓ^2-regularized algorithms, the SSR methods that utilize dictionary pruning, i.e., LPCA-SSR and $\text{SSR}_{\text{pruned}}$, performed notably better than the other block-sparse methods. LPCA-SSR performed better than $\text{SSR}_{\text{pruned}}$ (in terms of both classification accuracy and variance), though as we would expect, its advantage decreased as the sampling density increased. Further, neither method could match the performance of the algorithms that use ℓ^1-minimization, presumably because the pruned dictionary still allowed for a good approximation of the test sample in terms of each class separately. Note that if the dictionary pruning procedure were stricter or if the frequency of the sinusoidal waves was smaller (see the database description in Section 5.3), this might not be the case, and we would expect these methods to match or exceed their ℓ^1-based counterparts in terms of accuracy.

Figure 7.4 shows the results as the noise level η was varied on the synthetic database. Our observations are nearly-identical to those made of Figure 7.3, and so we do not restate them.

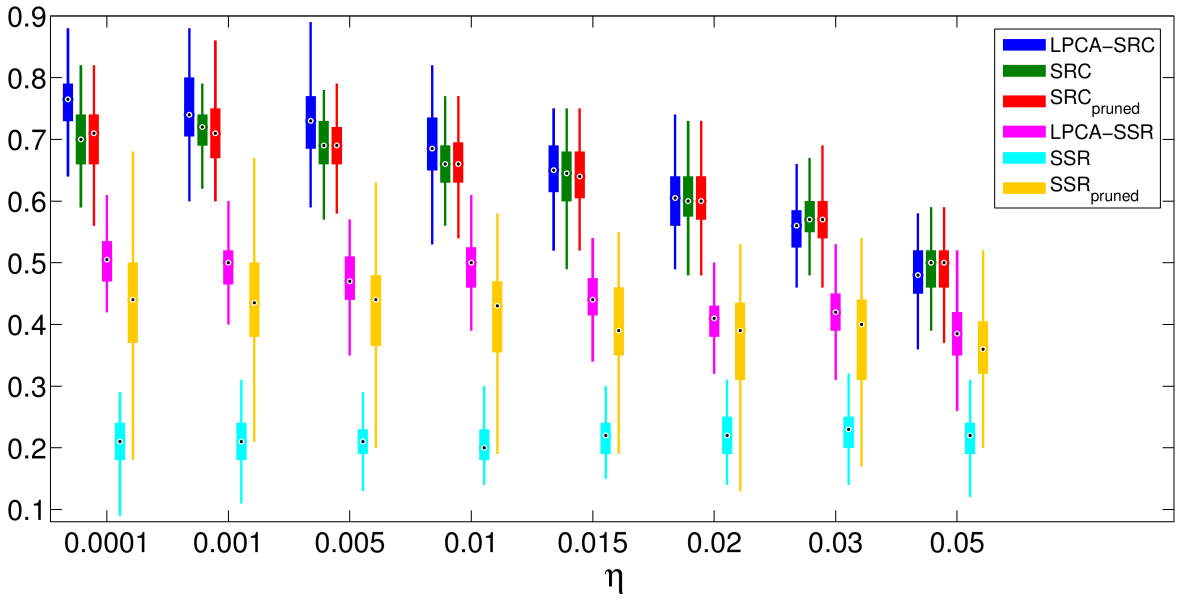


FIGURE 7.4. Box plot of the average classification accuracy (over 100 trials) of the original and SSR versions of algorithms on the synthetic database with varying noise level η. We fixed $N_0 = 25$.

To assess computational efficiency, the runtime of each algorithm on the synthetic database is displayed in Table 7.4 for every other value of N_0 and η. It is clear that, using SPGL1 on this database, the runtime to solve the BSR optimization problem scales poorly with N_{tr}, relative to the runtime of determining the ℓ^1-minimization solution.

	Varying N_0 ($\eta = 0.001$)				Varying η ($N_0 = 25$)			
Algorithm	$N_0 = 5$	$N_0 = 25$	$N_0 = 45$	$N_0 = 65$	$\eta = 0.001$	$\eta = 0.01$	$\eta = 0.02$	$\eta = 0.05$
LPCA-SRC	0.27	1.72	2.67	2.83	1.86	0.61	0.59	0.83
SRC	0.16	2.89	3.99	4.33	2.34	0.64	0.57	0.58
$\text{SRC}_{\text{pruned}}$	0.14	2.23	3.81	2.91	2.79	0.67	0.57	0.57
LPCA-SSR	0.55	5.89	11.82	18.35	5.94	1.93	3.03	3.13
SSR	0.17	2.35	6.02	9.49	2.15	5.36	5.64	3.79
$\text{SSR}_{\text{pruned}}$	0.25	2.01	7.76	12.90	3.33	4.06	4.56	3.38

TABLE 7.4. Average runtime (in seconds) of the original and SSR versions of algorithms over 100 trials on the synthetic database with varying training class size N_0 and varying noise level η.

We next consider the performance of the various algorithms on the face database AR-1. Table 7.5 contains the average accuracy, standard error, and runtime over 10 trials. We continue to see

the ℓ^1-based methods obtain higher classification accuracy than the methods that aim for block-sparsity. The latter algorithms improved as the dimension increased, though the ℓ^1-based methods still maintained the advantage at $m_{\text{PCA}} = 120$. This is discussed further throughout the rest of this chapter. With regard to runtime, it appears that the SSR methods scale much better with the feature dimension m_{PCA} than the training set size N_{tr} (for which poor scaling was observed on the synthetic database): the modified algorithms ran in roughly the same amount of time as the ℓ^1-based algorithms for all three values of m_{PCA}.

We also see, interestingly, that using SPGL1 in the LPCA-SRC and SRC algorithms produced slightly different results than using HOMOTOPY: LPCA-SRC had a slightly larger lead over SRC at $m_{\text{PCA}} = 30$ with SPGL1 (5% instead of 4%); however, SRC actually outperformed LPCA-SRC at $m_{\text{PCA}} = 120$ by 2%, instead of a 1% difference in the opposite direction using HOMOTOPY. This is likely the result of setting ϵ using cross-validation in SPGL1, whereas λ was fixed at 0.001 using HOMOTOPY on this database.

	$m_{\text{PCA}} = 30$			$m_{\text{PCA}} = 56$			$m_{\text{PCA}} = 120$		
Algorithm	Acc	SE	t	Acc	SE	t	Acc	SE	t
LPCA-SRC	**0.8499**	12.5	52.52	**0.9451**	6.8	108.33	0.9457	8.9	149.05
SRC	0.7969	5.6	63.35	0.9244	3.8	120.45	**0.9667**	2.5	189.97
$\text{SRC}_{\text{pruned}}$	0.8211	5.1	44.37	0.9320	2.5	77.34	0.9514	3.0	125.99
LPCA-SSR	0.7587	3.9	55.07	0.9020	4.4	102.90	0.9421	7.0	166.91
SSR	0.6516	6.2	60.12	0.8309	4.8	112.92	0.9276	3.2	157.20
$\text{SSR}_{\text{pruned}}$	0.7436	6.3	48.19	0.8946	9.0	77.08	0.9343	11.0	141.03

TABLE 7.5. Average accuracy, standard error ($\times 10^{-3}$), and runtime (in seconds) of the original and SSR versions of algorithms over 10 trials on AR-1.

Lastly, we consider the question of whether the optimization problem in Eq. (7.7) really recovered a block-sparse solution in the above experiments. Table 7.6 displays the average number of classes that contained training samples with nontrivial coefficients (i.e., coefficients with magnitude at least 0.001) in the solution vector $\boldsymbol{\alpha}^*$ for each of the six compared algorithms. We normalized this number by the total number of classes in the database.

We observe that the SSR methods were not, in general, more block-sparse than the methods that use ℓ^1-minimization, and they were sometimes less so. This is particularly clear in comparing SRC and SSR: the former algorithm was more block-sparse (sometimes significantly) at large values

	Synthetic ($\eta = 0.001$)				Synthetic ($N_0 = 25$)				AR-1		
Algorithm	$N_0 = 5$	25	45	65	$\eta = 0.001$	0.01	0.02	0.05	$m_{\text{PCA}} = 30$	56	120
LPCA-SRC	0.53	0.44	0.44	0.41	0.51	0.50	0.57	0.72	0.38	0.47	0.56
LPCA-SSR	0.69	0.58	0.58	0.47	0.69	0.51	0.52	0.64	0.35	0.50	0.51
SRC	0.58	0.39	0.39	0.32	0.59	0.53	0.56	0.74	0.63	0.91	1.00
SSR	0.47	0.46	0.46	0.65	0.45	0.75	0.86	0.84	0.61	0.95	1.00
$\text{SRC}_{\text{pruned}}$	0.56	0.39	0.39	0.40	0.52	0.54	0.58	0.76	0.50	0.60	0.61
$\text{SSR}_{\text{pruned}}$	0.49	0.50	0.50	0.45	0.57	0.62	0.62	0.66	0.45	0.55	0.58

TABLE 7.6. Average number of classes (divided by the total number of classes) with nontrivial coefficients in compared algorithms on the synthetic database (10 trials) and AR-1 (10 trials).

of both N_0 and η on the synthetic database, though both methods had similar block-sparsity on AR-1 and placed nontrivial coefficients at nearly all of the classes for $m_{\text{PCA}} = 56$ and at all of the classes for $m_{\text{PCA}} = 120$ on this data set. The methods $\text{SRC}_{\text{pruned}}$ and $\text{SSR}_{\text{pruned}}$ generally had about the same amount of block-sparsity; however $\text{SRC}_{\text{pruned}}$ was often slightly more block-sparse than $\text{SSR}_{\text{pruned}}$ on the synthetic database, and in contrast, $\text{SSR}_{\text{pruned}}$ was slightly block-sparser on AR-1. LPCA-SRC and LPCA-SSR also reported similar levels of block-sparsity, though the former algorithm was slightly more block-sparse on the synthetic database, except at large values of η for which the opposite was true. These differences were relatively small. The important observation is that we do not see convincing recovery of the block-sparsest solution via Eq. (7.7) in these results.

Further, as the values for AR-1 in Table 7.6 demonstrate, the degree of block-sparsity of each algorithm did not always correspond to its ability to obtain relatively high classification accuracy. Consider LPCA-SRC, for example, which had nontrivial coefficients at many fewer classes than SRC for $m_{\text{PCA}} = 120$, yet still the latter algorithm had higher classification accuracy at this feature dimension. Though our results rely on the threshold we used to define "nontrivial" coefficient values, this observation is perhaps important to explaining why ℓ^1-methods may sometimes work better than those based on BSR. This explanation, however, requires further investigation into block-sparse recovery theory: how close is the minimal mixed ℓ^2/ℓ^1-norm solution in Eq. (7.7) to the ground truth block-sparsest solution on these databases? Like the relationship between

SRC and the sparsest solution, we suspect that it may be difficult to verify block-sparse recovery conditions in these circumstances.[4]

7.3. Discussion

The synthetic database in notable in the context of representation-based algorithms in the sense that the corresponding system of equations is *very* underdetermined, with the set of training samples from any single class often able to produce a good approximation of the test sample. Geometrically, identifying the correct class comes down to isolating an *extremely* local neighborhood around $\boldsymbol{y}$. Neither ℓ^2-minimization nor block-sparsity provided a means of doing this. As in the case of LSDL-SRC in Chapter 4, the dictionary pruning step in LPCA-CRC, $\text{CRC}_{\text{pruned}}$, LPCA-SSR and $\text{SSR}_{\text{pruned}}$ managed to enforce some degree of locality, which improved classification performance. However, these methods could not match the effectiveness of ℓ^1-minimization, which achieved exceptionally local approximations of the test sample through its relationship to sparsity. Further, in the case of the SSR methods, the results in Table 7.6 suggest that minimization of the mixed ℓ^2/ℓ^1-norm was sometimes unable to recover the block-sparsest solution in this very underdetermined case.

On the face database AR-1, the negative correlation between the relative advantage of the ℓ^1-based methods and the feature dimension m_{PCA} is intriguing. On this database, ℓ^1-minimization (due to its relationship with sparsity) was critical only when m_{PCA} was small. In the case of the ℓ^2-regularized algorithms, this has been (imprecisely) explained in terms of the discriminate information contained in the test sample at large feature dimensions. The CRC authors Zhang et al. argued that ℓ^2-minimization is sufficient in these cases to produce a *naturally* sparse coefficient vector without the heavy machinery of ℓ^1-minimization [**115**]. Our experimental results in Table 7.3 suggest, however, that the ℓ^2-minimized coefficient vectors are *not* actually sparse, and that the diminishing value of ℓ^1-minimization as the feature dimension increases is due to the decreased redundancy in the dictionary resulting in a loss of sparsity in these algorithms.

With regard to block-sparse methods, we stress that these limited experiments on the synthetic database and AR-1 are insufficient to draw general conclusions about the efficacy of this approach

[4]In particular, we conjecture that this is the case in applications such as face recognition in which the classes themselves are highly-correlated. We discuss this correlation further in Chapter 11.

to classification. The particular class manifold structure of the synthetic database is clearly biased against methods that do not differentiate the training samples *within* each class, and it could be the case that we would see better block-sparse recovery in SSR on AR-1 at higher feature dimensions, under different pre-processing, when there are less classes, or when a different optimization algorithm is used. Again, full understanding of why ℓ^1-methods may perform better than BSR methods, as is the case of AR-1, involves investigation of how well real-world, class-structured data satisfies block-sparse recovery guarantees. This was not discussed in the SSR paper [**40**].

As a concluding remark, it would be informative to consider the relative performance of classifiers based on combinations of these methods. According to Hastie et al. [**49**], elastic-net regularization, which penalizes both the ℓ^1-norm and ℓ^2-norm of the coefficient vector, can improve performance in regression applications over ℓ^1-regularization alone. Elastic-net tends to return nonzero coefficients in "blocks" of correlated variables, i.e., selecting either all variables in the block or none of them [**49**]. Since Hastie et al.'s regression problem is formulated so that variables correspond to columns of the dictionary, this suggests that elastic-net could be used in representation-based classification to impose a (perhaps weak) block-sparse prior. When the blocks of highly-correlated samples correspond to classes, this method could be effective. Note that this is not a novel observation: see, for example, the paper by Timofte and Van Gool [**91**].

Methods that combine ℓ^1-minimization with BSR have also been proposed, which force the coefficient vector to be both block-sparse and sparse within each block. This approach is argued to, among other things, improve robustness. See, for example, Sprechmann et al.'s *Hierarchical Lasso* (HiLasso) [**88**], which has been applied to dictionary learning [**90**] and multiview facial recognition [**117**]. On the synthetic database especially, we suspect that this approach would be quite effective.

CHAPTER 8

Conclusion of Part 1

The main contribution of Part 1 was a modification of SRC called *local principal component analysis SRC*, or "LPCA-SRC." Through the use of tangent vectors, LPCA-SRC is designed to increase the sampling density of training sets and thus improve class discrimination in SRC on databases with sparsely-sampled and/or nonlinear class manifolds. The LPCA-SRC algorithm computes basis vectors of approximate tangent hyperplanes at the training samples in each class and replaces the dictionary of training samples in SRC with a local dictionary computed from shifted and scaled versions of these basis vectors and their corresponding training samples. Using a synthetic database and three face databases, we showed that LPCA-SRC can regularly achieve higher accuracy than SRC in cases of sparsely-sampled and/or nonlinear class manifolds, low noise, and relatively small PCA feature dimension.

Though the implementation details, particularly the setting of parameters, is somewhat involved in LPCA-SRC, the general concept of filling out class manifolds via tangent vectors is straightforward and easy to illustrate in low dimension. To address the issue of parameter setting, we suggested several viable methods for setting the class manifold dimension estimate d. It is important to note that in the case of small training sets, e.g., many face recognition problems, there are few options for the number-of-neighbors parameter n—and consequently for d by Eq. (4.3)—and so these values can easily be set using cross-validation, as in our experiments. When the training sets are very small (i.e., $N_l = 4$ or 5), one could simply set the parameters d and n to their maximum values, i.e., $d = n = N_{l_{\min}} - 2$, per Eq. (4.3). On the other hand, when algorithm efficiency is paramount, simply setting $d = 1$ may suffice.

One disadvantage of this method is its high computational cost. SRC is already expensive due to its ℓ^1-minimization procedure; in LPCA-SRC, the computation of tangent vectors is added to the algorithm's workload. The size of the dictionary in LPCA-SRC may be larger or smaller than that of SRC, depending on the LPCA-SRC parameters n and d and the effect of the pruning

parameter r. Thus LPCA-SRC can be slower or faster than SRC. As mentioned in Section 5.4.7, in deciding whether or not to use LPCA-SRC instead of SRC, simple computations based on the training data could be used to estimate the size of the pruned dictionary in LPCA-SRC: the user could compute the pruning parameter r, determine the number of training samples within r of $\boldsymbol{y}$, and then multiply this number by $d+1$. Knowledge of the dictionary size would give the user an indication of how the runtimes of LPCA-SRC and SRC would compare on the given data set.

Additionally, as we saw on the synthetic database, the usefulness of the tangent vectors in LPCA-SRC decreases as the noise level in the training data increases. This problem could likely be alleviated by using the method proposed by Kaslovsky and Meyer [**57**] to estimate clean points on the manifolds from noisy samples and then computing the tangent vectors at these points, as discussed in Chapter 6. Under the assumption that an approximately clean point is determined, we stated a theorem bounding (whp) the distance between an LPCA-SRC tangent vector and the corresponding class manifold, giving the user a potential tool to estimate the noise level of the computed tangent vectors when viewed as additional training samples. This information could then be factored into the decision of whether or not to use LPCA-SRC instead of SRC. Note that the case of large training sample noise was the only case for which we saw LPCA-SRC not obtain higher accuracy than SRC. Thus LPCA-SRC should be preferred over SRC in low noise scenarios on either small-scale problems (e.g., the size of ORL) or when achieving a modest (e.g., $1\% - 4\%$) boost in accuracy is worth potentially higher computational cost.

With regards to the *interpretability* (as discussed in Chapter 2) of LPCA-SRC, as for all representation-based classifiers, the position of large (in magnitude) coordinates of the coefficient vector $\boldsymbol{\alpha}^*$ can be used to identify training samples that are similar to the test sample, in the sense that they contribute substantially to its approximation. Additionally, the dictionary pruning technique in LPCA-SRC can give us some indication of the *separability* of the class manifolds via the average number of classes represented in the dictionary $D_{\boldsymbol{y}}$: if few, then this suggests that the class manifolds are fairly well-separated using Euclidean distance. Finally, visualization of the tangent vectors, as in Figures 5.4c, 5.5c, and 5.6c, can reveal details lost in PCA dimensionality reduction, as well as a small amount of information regarding the structure of each class manifold.

To test the generalization ability of our local PCA modification, we similarly used tangent vectors and dictionary pruning to amend CRC-RLS and Elhamifar and Vidal's SSR classification algorithm based on block sparsity. In both cases, our modification substantially improved performance. However, because the dictionary pruning step in our framework is performed online, LPCA-CRC was much slower than CRC-RLS (and no faster than the original LPCA-SRC), especially on large databases. Our experiments also showed that on the synthetic database and AR-1 for small feature dimensions, methods based on ℓ^1-minimization significantly outperformed their ℓ^2-regularized and block-sparse counterparts. In the case of the latter type of method, it could be that minimization of the mixed ℓ^2/ℓ^1-norm, used as an attempt at determining the block-sparsest solution, had poor recovery on these databases. However, given the motivation behind the BSR methods (in that they are *specifically* aimed at placing nonzero coefficients at training samples in the correct class), our empirical results should not discount BSR methods in general, especially those that simultaneously utilize ℓ^1-minimization.

Open questions regarding LPCA-SRC include whether or not the aforementioned general trends hold on other databases, in particular, in different classification applications. On the USPS database [**54**] for example, which contains samples of handwritten digits, LPCA-SRC does not offer an advantage over SRC, partially because significantly more energy is contained in the PCA feature vectors in USPS than those in the above face databases when the PCA dimension is the same.[1] Though LPCA-SRC is designed to increase the classification accuracy of SRC in the case of sparsely-sampled and/or nonlinear class manifolds, this algorithm requires a minimum sampling density (relative to the curvature) in order to obtain good tangent hyperplane estimates, as discussed in Chapter 6. Further, we do not expect it to perform well when the class manifolds are not smooth. On the other hand, if the class manifolds are *too* well-sampled with respect to curvature, tangent vectors are unneeded and LPCA-SRC will not offer an improvement over SRC. Since experimental results have indicated that handwritten-digit manifolds can be represented as lower-dimensional subspaces without losing too much information [**19**, **38**], we suspect that this is why LPCA-SRC does not outperform SRC on the well-sampled USPS database. However, since the class manifold

[1]For the curious reader, we note that LPCA-CRC and all three BSR methods were nearly as good (within roughly 2%) of the LPCA-SRC and SRC results on USPS. CRC was slightly worse, though it did offer an improvement over $\text{CRC}_{\text{unnorm.}}$ and $\text{CRC}_{\text{pruned}}$.

structure can only be roughly estimated (at best) in most real-world applications, it is difficult to predict the efficacy of LPCA-SRC in general.

Finally, it remains to investigate whether LPCA-SRC can improve upon SRC when different methods of dimensionality reduction besides PCA are used. One could also compare the performance of the per-class decomposition versions (in which test samples are approximated using class-specific dictionaries as discussed in Section 3.1) of the representation-based algorithms used in our experiments.

Part 2

Examining Sparsity in Classification

CHAPTER 9

Equivalence Guarantees

In the second part of this dissertation, we turn to the assumption on which SRC is fundamentally based: namely, that ℓ^1-minimization recovers a sparse solution. In this chapter, we review background material regarding so-called ℓ^1/ℓ^0-*equivalence* (referring to conditions under which minimization of the ℓ^1-norm recovers the same solution as minimization of the ℓ^0-"norm"), including the motivation behind this relationship and a brief tour of various equivalence theorems in this field.

9.1. Motivation from Compressed Sensing

Suppose that we wish to collect information about (i.e., sample or take measurements of) a continuous signal $f(t)$ and then send or store this information in an efficient manner. For example, $f(t)$ could be a sound wave or an image. Also suppose that a good approximation of the original signal must later be recovered. According to the Nyquist/Shannon sampling theorem, we must sample $f(t)$ at a rate of at least twice its maximum frequency in order to be able to reconstruct $f(t)$ exactly [**82**]. But in some applications, doing so may be expensive or even impossible.

In the circumstances that we are able to take many measurements of $f(t)$ to obtain its discrete analog $\boldsymbol{f} \in \mathbb{R}^N$, one efficient method of compressing it is the following procedure: Let the columns of $\Psi := [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_N]$ form an orthonormal basis for $\mathbb{R}^N$, and suppose that $\boldsymbol{f}$ has a sparse representation in this basis, i.e., that we can write $\boldsymbol{f} = \sum_{j=1}^N \alpha_j \boldsymbol{\psi}_j$, where $\alpha_j := \langle \boldsymbol{f}, \boldsymbol{\psi}_j \rangle$, $1 \leq j \leq N$, and $\boldsymbol{\alpha} := [\alpha_1, \ldots, \alpha_N]^\mathsf{T}$ is sparse. Setting all but the k largest (in absolute value) entries of $\boldsymbol{\alpha}$ to 0 in order to obtain $\boldsymbol{\alpha}_k$, it can be shown that $\Psi\boldsymbol{\alpha}_k$ gives the best k-term least squares approximation of $\boldsymbol{f}$ in this basis. Clearly, the sparser $\boldsymbol{\alpha}$ is, the better approximation we will obtain of $\boldsymbol{f}$, and in the case that $\|\boldsymbol{\alpha}\|_0 \leq k$, we recover the exact solution. This is the basic idea behind JPEG compression, which uses the discrete cosine transform as the sparsifying basis Ψ [**72**].

The problem with this procedure is that it is inefficient to spend time and money collecting all N samples if we are only going to throw most (all but k) of them away when the signal is compressed. This is the motivation behind *compressed sensing*, originally proposed by Candès and Tao [**14**] and Donoho [**28**] (see also Candès and Tao's work [**15**] and the paper by Candès et al. [**13**]). Let $\Phi \in \mathbb{R}^{m \times N}$ be a *sensing* or *measurement* matrix with $m < N$ and consider the underdetermined system

$$\boldsymbol{y}_0 := \Phi \boldsymbol{f} = \Phi \Psi \boldsymbol{\alpha} = X \boldsymbol{\alpha}$$

for sparse $\boldsymbol{\alpha}$, where we have set $X := \Phi\Psi$. Ideally, we would recover $\boldsymbol{f}$ by solving the optimization problem

$$\boldsymbol{\alpha}_0 := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \|\boldsymbol{\alpha}\|_0 \text{ subject to } X\boldsymbol{\alpha} = \boldsymbol{y}_0, \tag{9.1}$$

and set $\widehat{\boldsymbol{f}} := \Psi \boldsymbol{\alpha}_0$ with $\widehat{\boldsymbol{f}} \approx \boldsymbol{f}$. However, solving Eq. (9.1) is NP-hard. Fortunately, when X satisfies certain conditions and when $\boldsymbol{\alpha}_0$ is sufficiently sparse, the solution to Eq. (9.1) can be found by solving the ℓ^1-minimization problem

$$\boldsymbol{\alpha}_1 := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \|\boldsymbol{\alpha}\|_1 \text{ subject to } X\boldsymbol{\alpha} = \boldsymbol{y}_0. \tag{9.2}$$

This was a riveting finding, as the optimization problem in Eq. (9.2) is convex and can be solved efficiently. It has been shown that this procedure (under certain conditions, e.g., when the columns of Φ are uniformly random on the sphere S^{m-1}) produces an approximation of $\boldsymbol{f}$ that is as good as that of its best k-term approximation [**28**]. Further, theoretical and experimental results demonstrate that in many situations, the number of measurements m needed to recover $\boldsymbol{f}$ is significantly less than N and can be much lower than the number required by the Nyquist/Shannon theorem. For example, when the measurement matrix $\Phi \in \mathbb{R}^{m \times N}$ contains i.i.d. Gaussian entries, then exact recovery of $\boldsymbol{\alpha}_0$ via ℓ^1-minimization can be achieved (with high probability) in only $m = O(k \log(N/k))$ measurements, where $\|\boldsymbol{\alpha}_0\|_0 = k$ [**14**].

Even more astoundingly, similar results hold in the presence of noise, in which case the noiseless vector $\boldsymbol{y}_0$ is replaced with $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$, where $\boldsymbol{z} \in \mathbb{R}^m$ is a vector of errors satisfying $\|\boldsymbol{z}\|_2 \leq \epsilon$. Under certain conditions (involving how similar X is to an orthonormal basis when applied to

sufficiently-sparse vectors—see Definition 9.2.1), the ℓ^1-minimization problem

$$\boldsymbol{\alpha}_{1,\epsilon} := \arg\min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \|\boldsymbol{\alpha}\|_1 \text{ subject to } \|X\boldsymbol{\alpha} - \boldsymbol{y}\|_2 \leq \epsilon \tag{9.3}$$

is guaranteed to recover a coefficient vector approximating the ground truth sparse vector $\boldsymbol{\alpha}_0$ (the solution to Eq. (9.1)) with $\|\boldsymbol{\alpha}_{1,\epsilon} - \boldsymbol{\alpha}_0\|_2 \leq C_k\epsilon$ [**13**]. The constant C_k depends on properties of the matrix X.

A popular application of compressed sensing is *magnetic resonance imaging* (MRI), in which the measurement matrix Φ consists of m randomly-selected rows of the discrete Fourier transform in $\mathbb{R}^{N\times N}$ [**64**]. Other applications abound in the areas of data acquisition and compression, including sensor networks [**105**], seismology [**51**], and single pixel cameras [**34**].

The conditions under which the solutions to Eq. (9.1) and Eq. (9.2) are equal, i.e, ℓ^1/ℓ^0-equivalence holds, involve the number of nonzero coordinates of the sparsest solution $\boldsymbol{\alpha}_0$ and attributes of the matrix X. In the next section, we review several recovery guarantees, i.e., conditions guaranteeing exact or approximate recovery of the sparsest solution via ℓ^1-minimization. To make the problem more general, we no longer explicitly assume the use of a sparsifying transform matrix Ψ and consider the general system $X\boldsymbol{\alpha} = \boldsymbol{y}_0$, for $X \in \mathbb{R}^{m\times N}$ with $m < N$.

9.2. Review of Equivalence Guarantees

Existing results concerning ℓ^1/ℓ^0-equivalence consider the *incoherence* (or *spread*) of the vectors in the dictionary. Essentially, these results cannot be applied when the vectors are too correlated. A prototypical example is that if the data set contains two copies of the same vector (i.e., a pair of maximally-correlated vectors), then the minimum ℓ^1-norm solution may contain a nonzero coefficient at either one of the copies or at a combination of the two. Contrast this with the sparsest solution, which would never contain nonzero coefficients at both copies.

There are various ways of measuring the incoherence in a dictionary, each leading its own theory relating the solutions of Eq. (9.1) and Eq. (9.2) (or its noise version Eq. (9.3)). We review a few approaches here.

9.2.1. Restricted Isometry.

DEFINITION 9.2.1. *For any integer $k \in \{1, \ldots, N\}$, define the* isometry constant *of a matrix $X \in \mathbb{R}^{m \times N}$ as the smallest number δ_k such that*

$$(1 - \delta_k)\|\boldsymbol{\alpha}\|_2^2 \leq \|X\boldsymbol{\alpha}\|_2^2 \leq (1 + \delta_k)\|\boldsymbol{\alpha}\|_2^2$$

for all vectors $\boldsymbol{\alpha} \in \mathbb{R}^N$ satisfying $\|\boldsymbol{\alpha}\|_0 \leq k$.

Put simply, the isometry constant δ_k is a quantification of how close any set of k columns of X is to being an orthonormal basis. The following theorem uses the definition of isometry constants to relate the sparsest solution and that found by ℓ^1-minimization:

THEOREM 9.2.1 (Cai, Wang, and Xu [**9**], Theorem 3.2). *For $k > 1$, suppose that $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$, where $\|\boldsymbol{z}\|_2 \leq \epsilon$, $\boldsymbol{y}_0 = X\boldsymbol{\alpha}$, and $\|\boldsymbol{\alpha}\|_0 \leq k$. If the isometry constant δ_k of X satisfies $\delta_k < 0.307$, then*

$$\|\boldsymbol{\alpha}_{1,\epsilon} - \boldsymbol{\alpha}\|_2 \leq \frac{\epsilon}{0.307 - \delta_k},$$

where $\boldsymbol{\alpha}_{1,\epsilon}$ is the solution to Eq. (9.3)*.*

Note that since this holds for all $\boldsymbol{\alpha}$ satisfying $\boldsymbol{y}_0 = X\boldsymbol{\alpha}$ with $\|\boldsymbol{\alpha}\|_0 \leq k$, it must be that $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$ is the sparsest solution. In the case that $\epsilon = 0$, we have that $\boldsymbol{\alpha}_{1,\epsilon} = \boldsymbol{\alpha} = \boldsymbol{\alpha}_0$, and so ℓ^1/ℓ^0-equivalence holds! Further, this condition on δ_k is essentially tight; there exists a case for which ℓ^1/ℓ^0-equivalence does *not* hold and $\delta_k = (k-1)/(2k-1)$ [**9**]; considering that $(k-1)/(2k-1) = 1/3$ for $k = 2$ and $(k-1)/(2k-1) < 0.5$ for all $k \in \mathbb{N}$, this does not leave much room for improvement.

Random matrices, e.g., Gaussian and Bernoulli ensembles, have particularly small restricted isometry constants. For example, when the entries of X are drawn from $\mathcal{N}(0, 1/m)$, ℓ^1/ℓ^0-equivalence holds with overwhelmingly high probability provided that, for $\|\boldsymbol{\alpha}_0\|_0 \leq k$, the ratio k/N is sufficiently small. [**14**].

In the case that the coefficient vector $\boldsymbol{\alpha}$ is not sparse, the definition of restricted isometry can also be used to show that the ℓ^1-minimized solution is close to $\boldsymbol{\alpha}$, provided that $\boldsymbol{\alpha}$ has a good k-term approximation:

THEOREM 9.2.2 (Cai, Wang, and Xu [**9**], Theorem 3.3). *Given any noisy approximation* $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$ *with* $\boldsymbol{y}_0 = X\boldsymbol{\alpha}$*, where* $\|\boldsymbol{z}\|_2 \leq \epsilon$ *and the isometry constant* δ_k *of* X *satisfies* $\delta_k < 0.307$*, then*

$$\|\boldsymbol{\alpha}_{1,\epsilon} - \boldsymbol{\alpha}\|_2 \leq \frac{\epsilon}{0.307 - \delta_k} + \frac{1}{0.307 - \delta_k}\frac{\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_k\|_1}{\sqrt{k}},$$

where $\boldsymbol{\alpha}_{1,\epsilon}$ *is the solution to Eq.* (9.3) *and* $\boldsymbol{\alpha}_k$ *is the result of setting all but the* k *largest entries (in magnitude) of* $\boldsymbol{\alpha}$ *to 0.*

When $\boldsymbol{\alpha}$ satisfies $\|\boldsymbol{\alpha}\|_0 \leq k$, this reduces to Theorem 9.2.1.

Despite the wide applicability of this approach, computing the isometry constant to verify the conditions of the above theorems is often difficult. To calculate δ_k for a given X, we must examine all k-element subsets of the columns of X, which is expensive, prohibitively so when N is large. Unfortunately, this is also the case with the next notion of incoherence we examine.

9.2.2. Spark.

DEFINITION 9.2.2. *The* spark *of a matrix* X*, denoted* $\text{Spark}(X)$*, is the smallest number of linearly dependent columns of* X*.*

Though it initially appears quite similar to matrix rank, $\text{Spark}(X)$ is combinatorial to compute, requiring the examination of all increasingly large subsets of columns of X until a linearly-dependent subset is found. However, if it can be determined, the following theorem can be applied:

THEOREM 9.2.3 (Donoho and Elad [**30**], Corollary 4). *If a representation* $X\boldsymbol{\alpha} = \boldsymbol{y}_0$ *satisfies* $\|\boldsymbol{\alpha}\|_0 < \text{Spark}(X)/2$*, then* $\boldsymbol{\alpha}$ *is necessarily the sparsest such representation, i.e.,* $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$*, the solution to Eq.* (9.1)*.*

The relevant consequence of Theorem 9.2.3 is that if ℓ^1-minimization finds a coefficient vector $\boldsymbol{\alpha}_1$ (the solution to Eq. (9.2)) such that $\|\boldsymbol{\alpha}_1\|_0 < \text{Spark}(X)/2$, then ℓ^1/ℓ^0-equivalence holds. This bound is sharp in the sense that no quantity smaller than $\text{Spark}(X)/2$ can replace it in Theorem 9.2.3 [**30**].

To illustrate this theorem, consider that a random $m \times N$ matrix with full rank has spark equal to $m + 1$ with high probability, and so we only need $\|\boldsymbol{\alpha}_1\|_0 \leq (m + 1)/2$ to confirm that it is the sparsest solution.

Despite the difficulty in computing Spark(X), one can see that if there were a tractable lower bound on this quantity, Theorem 9.2.3 would become more useful. This brings us to the notion of *mutual coherence*.

9.2.3. Mutual Coherence.

DEFINITION 9.2.3. *Given a matrix $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{m \times N}$ with normalized columns (so that $\|\boldsymbol{x}_j\|_2 = 1$ for $1 \leq j \leq N$), the* mutual coherence *of X, denoted $\mu(X)$, is given by*

$$\mu(X) = \max_{1 \leq i \neq j \leq N} |\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle|.$$

Note that mutual coherence is straightforward to compute for a given dictionary. The relationship Spark(X) $\geq 1 + 1/\mu(X)$ [**8**, **30**] produces the following theorem:

THEOREM 9.2.4 (Donoho and Elad [**30**], Theorem 12; Gribonval and Nielsen [**46**], Theorem 1). *Let $X \in \mathbb{R}^{m \times N}$, $m < N$, have normalized columns and mutual coherence $\mu(X)$. If $\boldsymbol{\alpha}$ satisfies $X\boldsymbol{\alpha} = \boldsymbol{y}_0$ with*

$$\|\boldsymbol{\alpha}\|_0 < \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big), \tag{9.4}$$

then $\boldsymbol{\alpha}$ is the unique solution to the ℓ^1-minimization problem in Eq. (9.2).

Again, this means that if ℓ^1-minimization finds a solution with less than $(1/2)(1 + \mu(X)^{-1})$ nonzeros, then it is necessarily the sparsest solution and so ℓ^1/ℓ^0-equivalence holds.

Let us differentiate the noise tolerance ζ from the approximation error bound ϵ in Eq. (9.3), so that we can bound $\|\boldsymbol{z}\|_2 \leq \zeta$ and $\|X\boldsymbol{\alpha} - \boldsymbol{y}\|_2 \leq \epsilon$ separately.[1] (Note that $\zeta = \epsilon$ in Theorems 9.2.1 and 9.2.2.) Then the following theorem by Donoho et al. gives conditions for ℓ^1/ℓ^0-equivalence in the noisy setting:

THEOREM 9.2.5 (Donoho, Elad, and Temlyakov [**31**], Theorem 3.1). *Let $X \in \mathbb{R}^{m \times N}$, $m < N$, have normalized columns and mutual coherence $\mu(X)$. Suppose there exists an ideal noiseless signal*

[1]Given that $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$ with $\|\boldsymbol{z}\|_2 \leq \zeta$, we have that $\|X\boldsymbol{\alpha} - \boldsymbol{y}\|_2 \leq \|X\boldsymbol{\alpha} - \boldsymbol{y}_0\|_2 + \zeta$ for *any* coefficient vector $\boldsymbol{\alpha}$. This upper bound may be larger than ζ when $\boldsymbol{\alpha} \neq \boldsymbol{\alpha}_0$, and so we may set $\epsilon \geq \zeta$, as in the experiments in Chapter 11. However, since the bound $\|X\boldsymbol{\alpha} - \boldsymbol{y}\|_2 \leq \|X\boldsymbol{\alpha} - \boldsymbol{y}_0\| + \|\boldsymbol{z}\|_2$ could be very loose, we could also set $\epsilon < \zeta$.

$\boldsymbol{y}_0$ such that $\boldsymbol{y}_0 = X\boldsymbol{\alpha}$ and

$$\|\boldsymbol{\alpha}\|_0 = k \leq \frac{1}{4}\Big(1 + \frac{1}{\mu(X)}\Big). \tag{9.5}$$

Then $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$ is the unique sparsest representation of $\boldsymbol{y}_0$ over X. Further, suppose that we only observe $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$ with $\|\boldsymbol{z}\|_2 \leq \zeta$. Then we have

$$\|\boldsymbol{\alpha}_{1,\epsilon} - \boldsymbol{\alpha}_0\|_2^2 \leq \frac{(\epsilon + \zeta)^2}{1 - \mu(X)(4k-1)}, \tag{9.6}$$

where $\boldsymbol{\alpha}_{1,\epsilon}$ is the solution to Eq. (9.3)*.*

That is, if the ideal sparse vector $\boldsymbol{\alpha}_0$ is sparse enough and the mutual coherence of X is small enough, ℓ^1-minimization will give us a solution close to $\boldsymbol{\alpha}_0$, with "how close" depending on k, $\mu(X)$, the noise tolerance ζ, and the approximation error bound ϵ.

We can also say something regarding the support of $\boldsymbol{\alpha}_{1,\epsilon}$ in the noisy setting:

THEOREM 9.2.6 (Donoho, Elad, Temlyakov [**31**], Theorem 6.1). *Suppose that $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$, where $\boldsymbol{y}_0 = X\boldsymbol{\alpha}_0$, $\|\boldsymbol{\alpha}\|_0 \leq k$ and $\|\boldsymbol{z}\|_2 \leq \zeta$. Suppose that $\beta := \mu(X)k < \frac{1}{2}$ (so $k < \frac{1}{2\mu(X)}$). Set*

$$\gamma := \frac{\sqrt{1-\beta}}{1-2\beta}. \tag{9.7}$$

Then given $\boldsymbol{\alpha}_{1,\epsilon}$ the solution to Eq. (9.3) *with exaggerated error tolerance $\epsilon := C\zeta$ where $C = C(\mu(X), k) := \gamma\sqrt{k}$, we have that $\operatorname{supp}(\boldsymbol{\alpha}_{1,\epsilon}) \subset \operatorname{supp}(\boldsymbol{\alpha}_0)$.*

This says that when the mutual coherence is very small relative to the sparsity level, the solution $\boldsymbol{\alpha}_{1,\epsilon}$ to Eq. (9.3) has the same support as the sparsest solution $\boldsymbol{\alpha}_0$. (Observe that $\boldsymbol{\alpha}_0$ is indeed the sparsest solution by Theorem 9.2.4, since $\|\boldsymbol{\alpha}_0\|_0 < (1/2)\mu(X)^{-1} < (1/2)(1 + \mu(X)^{-1})$.) Since $\epsilon = \gamma\sqrt{k}\,\zeta$ and $\gamma \geq 1$, $\epsilon \geq \zeta$ is required in Theorem 9.2.6.

Despite the ease of computing mutual coherence, these theorems produce what are generally considered to be fairly loose bounds on the sparsity level k, given experimental results and cases for which restricted isometry constants are known [**49**]. As an example, consider that $\operatorname{Spark}(X)$ can be as large as $m + 1$, whereas $1 + \mu(X)^{-1}$ is bounded above by $1 + \sqrt{m}$, assuming that the columns of X are normalized. Thus we expect that Theorem 9.2.3 will generally allow us to take many less measurements (and still prove ℓ^1/ℓ^0-equivalence) than Theorem 9.2.4.

There are other recovery guarantees involving incoherence that hold with high probability. When applied to random matrices, these guarantees are generally stronger than those discussed above (in terms of requiring less measurements and/or less sparsity of the solution vector). For example, Candès and Plan [**11**] provide conditions that guarantee recovery (with high probability) of sparse and approximately sparse solutions in the case that the rows of the dictionary are sampled independently from certain probability distributions. These conditions are in terms of incoherence, defined as an upper bound on the squared norms of the rows of X (either deterministically or stochastically), and require an *isotropy* property [**11**]. In the case that the probability distribution has mean 0, this property states that the covariance matrix of the probability distribution is equal to the identity matrix.

In another paper [**10**], Candès and Plan guarantee probabilistic recovery in terms of a condition on mutual coherence (as defined in Definition 9.2.3) that is satisfied with high probability on certain random matrices. These recovery guarantees allow for the sparsity level k in the case of these random matrices to be notably larger than in Eq. (9.4) in Theorem 9.2.4. We also mention the results by Tropp [**92**] concerning recovery in terms of mutual coherence and the extreme singular values of randomly-chosen subsets of dictionary columns.

9.2.4. Neighborliness. Thus far, the equivalence guarantees we have looked at have given only *sufficient* conditions for ℓ^1/ℓ^0-recovery. The following notion of incoherence provides a condition that is also *necessary*. Define P, the polytope associated with the matrix $X \in \mathbb{R}^{m\times N}$, as $P := XC$, where C is the N-dimensional polytope determined by $C := \{\boldsymbol{x} \in \mathbb{R}^N : \|\boldsymbol{x}\|_1 \leq 1\}$.

Definition 9.2.4. *The polytope P associated with the matrix X is said to be* k-neighborly *if every set of $k+1$ vertices (that does not include an antipodal pair) forms a face of P.*

Theorem 9.2.7 (Donoho [**27**], Theorem 1 (see also Donoho and Tanner [**32**])). *The polytope P associated with the matrix X has $2N$ vertices and is k-neighborly if and only if whenever the sparsest solution $\boldsymbol{\alpha}_0$ satisfying $X\boldsymbol{\alpha}_0 = \boldsymbol{y}$ has no more than k nonzeros, ℓ^1-minimization in the form of Eq. (9.2) recovers $\boldsymbol{\alpha}_0$.*

That is, the associated polytope having $2N$ vertices and being k-neighborly is a necessary and sufficient condition for ℓ^1/ℓ^0-equivalence in the case that $\|\boldsymbol{\alpha}_0\|_0 \leq k$. In general, polytopes of the

form P are no more than $k \leq \lceil (m+1)/3 \rceil$ neighborly [**27**]. Combining this fact with Theorem 9.2.7 produces the following corollary:

COROLLARY 9.2.1 (Donoho [**27**], Corollary 1.3 (see also Donoho and Tanner [**32**])). *For $N-2 \geq m > 2$, if ℓ^1-minimization in the form of Eq.* (9.2) *recovers all sparse vectors satisfying $\|\boldsymbol{\alpha}\|_0 \leq k$, then $k \leq \lceil (m+1)/3 \rceil$.*

Unfortunately, determining whether P is k-neighborly is not tractable for large dimension m. In the SRC paper [**104**], Wright et al. used a combinatorial (but relatively efficient) algorithm [**83**] to compute a tighter upper bound on polytope neighborliness and used this bound to estimate for which dimension (i.e., number of measurements) m ℓ^1/ℓ^0-equivalence can be achieved. However, this is a very imprecise approach, as an upper bound on neighborliness can only provide us with a negative result of the following form: If P is no more than k'-neighborly and $\|\boldsymbol{\alpha}_1\|_0 := k > k'$, then $\boldsymbol{\alpha}_1 \neq \boldsymbol{\alpha}_0$, i.e., ℓ^1/ℓ^0-equivalence does not hold. However, in the case that $k \leq k'$, nothing more can be said.

CHAPTER 10

Mutual Coherence Equivalence in the Context of Classification

In classification problems, data from the same class may be highly-correlated. As we will show, the ℓ^1/ℓ^0-recovery guarantees that can be applied in practice often do not hold. Thus the "theory" behind sparse representation-based methods for classification is missing a significant piece. Though such methods work experimentally well, is their success really due to sparsity? Is sparsity really being achieved? We aim to provide insight into these questions via a series of projects in the next few chapters.

Though in the last chapter we used $\boldsymbol{y}_0$ to refer to a clean measurement vector and $\boldsymbol{y} := \boldsymbol{y}_0 + \boldsymbol{z}$ to refer to its noisy version, in this chapter and in Chapter 12, $\boldsymbol{y}$ may represent *either* a clean or noisy measurement vector, or more commonly, an arbitrary test sample (as was done in Part 1). We do this because, in the context of representation-based classification, there are reasons other than noise in the test sample for allowing the equality $\boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}$ to hold only approximately: the training data could also be corrupted, or we may want to relax the assumption that class manifolds are linear subspaces (perhaps this is only approximately, or locally, the case). To keep the situation general and to avoid confusion, we we will only differentiate between $\boldsymbol{y}$ and $\boldsymbol{y}_0$ when we explicitly consider $\boldsymbol{z}$ as noise, as in Donoho et al.'s Theorems 9.2.5 and 9.2.6 that consider mutual coherence in the case of noise.

That said, it is important to note that in the case that the equality $\boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}$ holds only approximately due to class manifold curvature, theorems such as Theorem 9.2.5 can still (in theory) be applied to show that the solution found by ℓ^1-minimization approximates the sparsest solution. Here, the vector $\boldsymbol{z}$ can refer to the error between a *clean* test sample $\boldsymbol{y}$ and the projection of $\boldsymbol{y}$ onto the span of (a portion of) its same-class training samples. In this case, the notation $\hat{\boldsymbol{y}}_l$ for the projection of a class l test sample $\boldsymbol{y}$ onto the span of local class l training samples, with $\boldsymbol{y} = \hat{\boldsymbol{y}}_l + \boldsymbol{z}$, is perhaps more appropriate than using $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{z}$. This is another reason to generalize our notation.

10.1. The Dilemma

In this chapter, we investigate whether or not ℓ^1/ℓ^0-recovery guarantees involving mutual coherence are compatible with the assumptions made in SRC. As we saw in the last chapter, this type of recovery guarantee is easily computable, unlike those involving restricted isometry constants, spark, or polytope neighborliness.

Ideally, we could use the mutual coherence conditions given in Theorem 9.2.4 or 9.2.5 to show that ℓ^1-minimization in SRC produces a solution equal or close to the sparsest solution. However, in classification problems, samples in the same class can be extremely correlated, resulting in $\mu(X_{\text{tr}})$ being close to 1 (we assume that the training samples have ℓ^2-norm equal to 1, throughout). For example, recall the sinusoidal wave synthetic database used in Chapter 5: With training class size $N_0 = 25$ and noise level $\eta = 0.001$, the average mutual coherence of X_{tr} over 100 trials is 1.0000. As further evidence of the large mutual coherence inherent in classification problems, Table 10.1 displays the average values of $\mu(X_{\text{tr}})$ (over 10 trials) of the face databases used in Chapter 5. The training sets reflect 50/50 training/test splits.

Database	$m_{\text{PCA}} = 30$	$m_{\text{PCA}} = 56$	$m_{\text{PCA}} = 120$
AR-1	0.9991	0.9987	0.9985
AR-2	0.9993	0.9988	0.9984
Ext Yale B	0.9951	0.9954	0.9941
ORL	0.9971	0.9970	0.9966

TABLE 10.1. Average mutual coherence (over 10 trials) computed from the samples in X_{tr} (50/50 training/test splits) of the face databases from Chapter 5.

When $\mu(X_{\text{tr}}) \approx 1$, the mutual coherence bound in Theorem 9.2.4 becomes

$$\|\boldsymbol{\alpha}\|_0 < \frac{1}{2}\Big(1 + \frac{1}{\mu(X_{\text{tr}})}\Big) \approx 1.$$

Since $\|\boldsymbol{\alpha}\|_0$ denotes the number of nonzero coefficients in the representation of $\boldsymbol{y}$ over X_{tr}, it will never satisfy $\|\boldsymbol{\alpha}\|_0 < 1$. Thus we cannot use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence in SRC, for example, on the databases used in Chapter 5.

10.2. Main Goal

In the following sections, we identify cases in which the condition given in Eq. (9.4) from Theorem 9.2.4 *provably* does not hold, and thus we cannot use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence. We also discuss analogous results in the noisy case, i.e., Eq. (9.5) in Theorem 9.2.5. In particular, we are concerned with the applicability of these theorems for class-structured data.

10.3. Preliminary Results

We will use the following lemma which gives a lower-bound on mutual coherence in the underdetermined setting:

LEMMA 10.3.1 (Rosenfeld [**77**], Theorem 3). *For $X \in \mathbb{R}^{m\times N}$ with normalized columns and $m < N$, we have that*

$$\mu(X) \geq \sqrt{\frac{N-m}{m(N-1)}}.$$

It is straightforward to show that Lemma 10.3.1 implies that

$$\mu(X) \geq \frac{1}{m},$$

since $\sqrt{\frac{N-m}{m(N-1)}}$ monotonically increases in $N \in \mathbb{N}$ for $N > m$, with a minimum value of $1/m$ attained at $N = m + 1$. Thus to have even a *chance* of Theorem 9.2.4 or 9.2.5 holding, we must have

$$\|\boldsymbol{\alpha}\|_0 < \frac{1}{c}\Big(1 + \frac{1}{\mu(X)}\Big) \leq \frac{1}{c}\Big(1 + m\Big),$$

where $c = 2$ in the noiseless case and $c = 4$ in the noisy setting.

We next consider the smallest possible value of the number of nonzeros $\|\boldsymbol{\alpha}\|_0$ in any representation $X\boldsymbol{\alpha} = \boldsymbol{y}$. Let us assume that each test sample is not a scalar multiple of any training sample. It follows that $\|\boldsymbol{\alpha}\|_0$ is at least 2. Thus in order for Theorem 9.2.4 or 9.2.5 to hold, we must have

$$2 \leq \|\boldsymbol{\alpha}\|_0 < \frac{1}{c}\Big(1 + \frac{1}{\mu(X_{\text{tr}})}\Big) \Rightarrow \mu(X_{\text{tr}}) < \frac{1}{2c-1}$$

$$\Rightarrow \mu(X_{\text{tr}}) < \begin{cases} 1/3, & \text{noiseless case} \\ 1/7, & \text{noisy setting.} \end{cases}$$

Note that these upper bounds for $\mu(X_{\text{tr}})$ are very small compared to the values of $\mu(X_{\text{tr}})$ in Table 10.1!

These findings produce the following small-scale result:

PROPOSITION 10.3.1. *Suppose that $X_{\text{tr}}\boldsymbol{\alpha} = \boldsymbol{y}$. If $m \leq 3$ and $\boldsymbol{y}$ is not a scalar multiple of any training sample, then the inequality in Eq.* (9.4) *with $X = X_{\text{tr}}$ does not hold. That is, we cannot use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence in SRC.*

Proof. By Lemma 10.3.1, we must have that $\mu(X_{\text{tr}}) \geq \frac{1}{m} \geq \frac{1}{3}$. □

An analogous statement holds in the noisy setting (Theorem 9.2.5) for $m \leq 7$.

10.4. Main Result

PROPOSITION 10.4.1 (Main Result). *Suppose that the sparsest representation of $\boldsymbol{y} \in \mathbb{R}^m$ over the dictionary $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{m \times N}$ is given by $\boldsymbol{y} = \alpha_1 \boldsymbol{x}_{j_1} + \ldots + \alpha_{j_k} \boldsymbol{x}_{j_k}$ for $\{j_1, \ldots, j_k\} \subset \{1, \ldots, N\}$. Set $\widetilde{N}$ to be the number of columns of X contained in*

$$\widetilde{\mathcal{X}} := \text{span}\{\boldsymbol{x}_{j_1}, \ldots, \boldsymbol{x}_{j_k}\}.$$

Of course, $\widetilde{N} \geq k$. If $\widetilde{N} > k$, then the inequality in Eq. (9.4) *does not hold. That is, we cannot use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence.*

Proof: Suppose that $\widetilde{N} > k$. Then there are more than k dictionary elements in the subspace $\widetilde{\mathcal{X}}$. Since the vectors $\boldsymbol{x}_{j_1}, \ldots, \boldsymbol{x}_{j_k}$ are linearly independent (because otherwise, $\boldsymbol{y}$ could be expressed more sparsely), the dimension of $\widetilde{\mathcal{X}}$ is exactly k.

Define $\widetilde{X} \in \mathbb{R}^{m \times \widetilde{N}}$ to be the matrix of the $\widetilde{N}$ dictionary elements contained in $\widetilde{\mathcal{X}}$. Let the singular value decomposition of $\widetilde{X}$ be given by $\widetilde{X} = U\Sigma V^{\mathsf{T}}$, and set U_k to contain the first k columns of U, V_k to contain the first k columns of V, and Σ_k to contain the first k columns and rows of Σ. Because $\widetilde{X}$ has rank k, we can alternatively write

$$\widetilde{X} = U_k \Sigma_k V_k^{\mathsf{T}}.$$

The $k \times \widetilde{N}$ matrix $U_k^\mathsf{T} \widetilde{X}$ has the same mutual coherence as $\widetilde{X}$, since they have the same Gram matrices:

$$\begin{aligned}(U_k^\mathsf{T} \widetilde{X})^\mathsf{T}(U_k^\mathsf{T} \widetilde{X}) &= \widetilde{X}^\mathsf{T} U_k U_k^\mathsf{T} \widetilde{X} \\ &= (U_k \Sigma_k V_k^\mathsf{T})^\mathsf{T} U_k U_k^\mathsf{T} (U_k \Sigma_k V_k^\mathsf{T}) \\ &= V_k \Sigma_k^\mathsf{T} U_k^\mathsf{T} U_k U_k^\mathsf{T} U_k \Sigma_k V_k^\mathsf{T} \\ &= V_k \Sigma_k^\mathsf{T} \Sigma_k V_k^\mathsf{T} \\ &= (U_k \Sigma_k V_k^\mathsf{T})^\mathsf{T} (U_k \Sigma_k V_k^\mathsf{T}) \\ &= \widetilde{X}^\mathsf{T} \widetilde{X}.\end{aligned}$$

By Lemma 10.3.1, we have that

$$\mu(X) \geq \mu(\widetilde{X}) = \mu(U_k^\mathsf{T} \widetilde{X}) \geq \sqrt{\frac{\widetilde{N} - k}{k(\widetilde{N} - 1)}} \geq \sqrt{\frac{(k+1) - k}{k((k+1) - 1)}} = \frac{1}{k}.$$

Thus the bound on k in Theorem 9.2.4 requires that

$$k < \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big) \leq \frac{1}{2}(1 + k) \Rightarrow k < 1. \quad \square \tag{10.1}$$

We present several corollaries to Proposition 10.4.1. The first is a consequence applicable to any ℓ^1-minimization problem, regardless of whether or not the dictionary elements have class structure:

COROLLARY 10.4.1 (Consequence for general ℓ^1-minimization). *If a measurement vector $\boldsymbol{y} \in \mathbb{R}^m$ is not at all sparse over the dictionary $X \in \mathbb{R}^{m \times N}$, i.e., if every representation of $\boldsymbol{y}$ requires no less than m dictionary elements, then the condition in Eq.* (9.4) *from Theorem 9.2.4 does not hold. This is because the dimension of $\widetilde{\mathcal{X}}$ (as defined in Proposition 10.4.1) is m, and so every dictionary element is contained in $\widetilde{\mathcal{X}}$.*

Corollary 10.4.1 illustrates the importance of choosing a dictionary that awards a sparse representation of $\boldsymbol{y}$ in any application of ℓ^1-minimization.

The following corollary follows from the proof of Proposition 10.4.1:

COROLLARY 10.4.2. *Let $X \in \mathbb{R}^{m\times N}$ with $m < N$, and let k be any positive integer such that $k < N$. If* any *set of k linearly independent columns of X spans an additional, distinct column of X, then the bound*

$$k < \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big)$$

does not hold.

Of course, this bound will not hold for any larger values of k, either. This means that if we can find an integer k satisfying the conditions of Corollary 10.4.2, then any attempt to prove ℓ^1/ℓ^0-equivalence using Theorem 9.2.4 will require $X\boldsymbol{\alpha} = \boldsymbol{y}$ with $\|\boldsymbol{\alpha}\|_0 < k$. Recalling the previous chapter, the hypothesis of Corollary 10.4.2 states that $\text{Spark}(X) \leq k + 1$. Using that $\text{Spark}(X) \geq 1 + 1/\mu(X)$ [**8**, **30**], we can alternatively prove Corollary 10.4.2 by noting that

$$k \geq \frac{k+1}{2} \geq \frac{\text{Spark}(X)}{2} \geq \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big).$$

Though computing $\text{Spark}(X)$ is generally not tractable, Corollary 10.4.2 only requires computing an upper bound for $\text{Spark}(X)$. Further, if $X = X_{\text{tr}}$ is the training set in a classification problem, we might expect that $\text{Spark}(X_{\text{tr}})$ is lower than that for arbitrary dictionaries due to the dictionary's class structure. Thus we may only need to try a few small values for k in order to apply Corollary 10.4.2.

The following corollary is an explicit consequence for class-structured dictionaries:

COROLLARY 10.4.3 (Consequence for Class-Structured Dictionaries). *Suppose that $\boldsymbol{y}$ is a test sample with $\|\boldsymbol{y}\|_2 = 1$, and define $\mu := \mu(X_{\text{tr}})$. If the addition of $\boldsymbol{y}$ does not increase the mutual coherence, that is, if $|\langle \boldsymbol{y}, \boldsymbol{x}_i\rangle| \leq \mu$ for all $1 \leq i \leq N_{\text{tr}}$, i.e., $\mu([\boldsymbol{y}, X_{\text{tr}}]) = \mu$, then we cannot have both that (i) $X_{\text{tr}}\boldsymbol{\alpha} = \boldsymbol{y}$ and (ii) $\|\boldsymbol{\alpha}\|_0 < (1/2)(1 + (1/\mu(X_{\text{tr}})))$.*

Proof: If we can write $X_{\text{tr}}\boldsymbol{\alpha} = \boldsymbol{y}$ for $\|\boldsymbol{\alpha}\|_0 =: k$, then the k (linearly independent) training samples with nonzero coefficients in the representation span a k-dimensional subspace containing $\boldsymbol{y}$. Setting $X = [\boldsymbol{y}, X_{\text{tr}}]$ in Corollary 10.4.2, we have that

$$k \not< \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big) = \frac{1}{2}\Big(1 + \frac{1}{\mu}\Big).$$

On the other hand, if

$$k < \frac{1}{2}\Big(1 + \frac{1}{\mu(X)}\Big) = \frac{1}{2}\Big(1 + \frac{1}{\mu}\Big)$$

for some positive integer $k < N_{\text{tr}}$, then also by Corollary 10.4.2, it must be the case that $\boldsymbol{y}$ is not contained in the subspace spanned by any k linearly independent distinct columns of X, i.e., columns of X_{tr}. Thus we cannot write $X_{\text{tr}}\boldsymbol{\alpha} = \boldsymbol{y}$ for any $\boldsymbol{\alpha}$ satisfying $\|\boldsymbol{\alpha}\|_0 = k$. $\square$

It might initially seem that the hypothesis of Corollary 10.4.3 is unlikely to hold. However, if one assumes that the data is sampled randomly with test samples having the same distribution as the training samples in their ground truth classes, then the hypothesis that $\mu([\boldsymbol{y}, X_{\text{tr}}]) = \mu(X_{\text{tr}})$ becomes much more probable. This actually becomes a critical problem for us in Chapter 12.

Our final corollary determines conditions under which the bound in Eq. (9.4) from Theorem 9.2.4 is theoretically incompatible with the explicit assumptions made in SRC. We review these assumptions briefly:

Assumption 1 (Linear Subspaces). *The ground truth class manifolds of the given data set are linear subspaces.*

Assumption 2 (Spanning Training Set). *The training matrix X_{tr} contains sufficient samples in each class to span the corresponding linear subspace.*

Corollary 10.4.4 (Consequence for SRC). *Suppose that the SRC Assumptions 1 and 2 hold. Let $\boldsymbol{y}$ have ground truth class l, and suppose that the number of class l training samples, N_l, is large, i.e., $N_l > d_l$, for d_l the dimension of the linear subspace representing the class l manifold. Then there exists a test sample $\boldsymbol{y}$ which requires the maximum number d_l of class l training samples to represent it. If this representation of $\boldsymbol{y}$ is its sparsest representation over the dictionary X_{tr}, then the condition in Eq.* (9.4) *from Theorem 9.2.4 cannot hold. Thus we cannot use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence in SRC.*

Corollary 10.4.4 says that if we have a surplus of class l training samples (i.e., more than enough to span the class l subspace), then, provided that the "class representations" (representations of the test samples in terms of their ground truth classes) truly are the sparsest representations of

the test samples over the training set (as argued by the SRC authors [**104**]), there will be some test samples for which Theorem 9.2.4 cannot hold. These test samples are exactly those requiring $k = d_l$ class l training samples in their representations. In general, such test samples must exist; otherwise, the dimension of the class l subspace would be less than d_l. To reiterate, if everything we *want* to happen in SRC happens (large class sizes, sparse class representations), then we cannot consistently use Theorem 9.2.4 to prove ℓ^1/ℓ^0-equivalence.

On a more positive note, the assumptions in SRC make it possible to estimate whether or not the conditions of Proposition 10.4.1 hold. Though these conditions are difficult to check in general (if we knew the sparsest solution of $\boldsymbol{y}$ over the dictionary, then we would not need to use ℓ^1-minimization to find it!), the linear subspace assumption in SRC gives us a heuristic for doing so. We could potentially estimate the dimension of each class (using one of the methods discussed in the context of the LPCA-SRC algorithm in Chapter 4, for instance) and compare this with the number of training samples in that class. If the latter is larger than the former, then we expect that Theorem 9.2.4 cannot be applied for some test samples.

In typical applications, we must deal with noisy data. Thus we should consider the application of Theorem 9.2.5 instead of Theorem 9.2.4. But this is immediate: Since the mutual coherence condition is stricter in the case of noise, the consequences of Proposition 10.4.1 and the above corollaries hold whenever the conditions are assumed to hold on the clean version of the data. In particular, Theorem 9.2.5 requires the *existence* of a clean test sample $\boldsymbol{y}_0$ (even if it is unknown to us) that satisfies $X\boldsymbol{\alpha} = \boldsymbol{y}_0$ with $\|\boldsymbol{\alpha}\|_0 \leq (1/4)(1 + (1/\mu(X)))$. Under the hypothesis of Corollary 10.4.3 (setting $\boldsymbol{y}_0 = \boldsymbol{y}$), such a $\boldsymbol{y}_0$ cannot exist. An analogous result holds in the case that class manifolds contain curvature, using the framework discussed in the beginning of this chapter.

As a concluding remark, we stress that the mutual coherence conditions in Theorems 9.2.4 and 9.2.5 are sufficient, but not necessary, for ℓ^1/ℓ^0-equivalence. Thus it is possible for ℓ^1-minimization to find (or closely approximate) the sparsest solution even when the conditions of these theorems do not hold. Whether or not this happens in the context of SRC is the topic of the next chapter.

CHAPTER 11

Equivalence on Highly-Coherent Data

In this chapter, we investigate whether sparsity is reliably achieved via ℓ^1-minimization on highly-correlated data, such as class-structured databases.

11.1. Inspiration

We are inspired by the data model and subsequent work of Wright and Ma [**102**] (see also the work of Wright et al. [**103**]), which produces an ℓ^1/ℓ^0-equivalence guarantee for dictionaries containing vectors assumed to model facial images. We summarize their result briefly.

Previous work has shown that the set of facial images of a fixed subject (person) under varying illumination conditions forms a convex cone, called an *illumination cone*, in pixel space [**3**, **44**]. Wright and Ma demonstrate that in fact the set of facial images under varying illuminations over *all subjects combined* exhibits this cone structure. For example, they show that this is the case for the entire set of (raw) samples from the Extended Yale B Face Database [**44**]. Further, this cone becomes extremely narrow, i.e., a "bouquet," as the number of pixels grows large [**102**]. These findings reiterate that class-structured data, particularly face databases, are highly-coherent.

Lee et al. [**59**] showed that any image from the illumination cone can be expressed as a linear combination of just a few images of the same subject under varying lighting conditions. In other words, illumination cones are well-approximated by linear subspaces. Thus the SRC condition that class manifolds are (approximately) linear subspaces presumably holds for databases made up of facial images under varying lighting conditions. Given a facial image $\boldsymbol{y} \in \mathbb{R}^m$ that may be occluded or corrupted by noise, $\boldsymbol{y}$ can thus be expressed as

$$\boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha}_0 + \boldsymbol{z}_0, \tag{11.1}$$

given that certain requirements are satisfied in the sampling of the training data. By the above model, $\boldsymbol{\alpha}_0$ is assumed to be non-negative (a result of the illumination cone model [**44**, **103**]) and

sparse, containing nonzeros at training samples that represent the same subject as $\boldsymbol{y}$ (i.e., are in the same class). Additionally, $\boldsymbol{z}_0$ is an (unknown) error vector with nonzeros in only a fraction of its coordinates; i.e., the model assumes that only a portion of the pixels are occluded or corrupted [**103**].

The goal, as one might expect, is to recover $\boldsymbol{\alpha}_0$ from Eq. (11.1). In the SRC paper [**104**], Wright et al. use ℓ^1-minimization to do this. In particular, they solve

$$(\boldsymbol{\alpha}_1, \boldsymbol{z}_1) := \arg\min \|\boldsymbol{\alpha}\|_1 + \|\boldsymbol{z}\|_1 \text{ subject to } \boldsymbol{y} = X_{\text{tr}}\boldsymbol{\alpha} + \boldsymbol{z}, \tag{11.2}$$

and they show that this "occlusion version" of SRC (previously mentioned in Section 5.4.3) produces very good classification results on corrupted facial images.

In a later paper, Wright, et al. [**103**] correctly note that the usual ℓ^1/ℓ^0-equivalence theorems do not hold on the highly-correlated data in X_{tr}, and so we are unable to determine whether or not the ℓ^1-minimized solution $\boldsymbol{\alpha}_1$ in Eq. (11.2) is equal to (what is assumed to be) the true sparsest solution $\boldsymbol{\alpha}_0$. Fortunately, Wright and Ma [**102**] proved a theorem that gives sufficient conditions for this equivalence under an assumed model (called the *bouquet model*) of facial images; see also Wright et al.'s version [**103**]. To state the theorem, we will need the following definition:

DEFINITION 11.1.1 (Proportional Growth [**102**]). *A sequence of signal-error problems* $\boldsymbol{y} = X\boldsymbol{\alpha}_0 + \boldsymbol{z}_0$*, for* $X \in \mathbb{R}^{m\times N}$*, exhibits* proportional growth *with parameters* $\delta > 0$*,* $\rho \in (0, 1)$*, and* $\beta > 0$*, if* $N = \lfloor \delta m \rfloor$*,* $\|\boldsymbol{z}_0\|_0 = \lfloor \rho m \rfloor$*, and* $\|\boldsymbol{\alpha}_0\|_0 = \lfloor \beta m \rfloor$*.*

It follows that δ is the redundancy factor in the dictionary X and ρ and β control the sparsity of $\boldsymbol{z}_0$ and $\boldsymbol{\alpha}_0$, respectively. Here, β is assumed to be small and may depend on δ and ρ.

We are now in a position to state Wright and Ma's main theorem:

THEOREM 11.1.1 (Wright and Ma [**102**], Theorem 1). *Fix any* $\delta > 0$ *and* $\rho < 1$*. Suppose that* X *is distributed according to the* bouquet model *given by*

$$X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{m\times N},\ \boldsymbol{x}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{\mu}, (\nu^2/m)I_m),\ \|\boldsymbol{\mu}\|_2 = 1,\ \|\boldsymbol{\mu}\|_\infty \leq C_\mu m^{-1/2},\ C_\mu \geq 1 \tag{11.3}$$

for ν *sufficiently small. Also suppose that the sequence of signal-error problems* $\boldsymbol{y} = X\boldsymbol{\alpha}_0 + \boldsymbol{z}_0$ *for* $X \in \mathbb{R}^{m\times N}$ *exhibits proportional growth with parameters* δ*,* ρ*, and* β*. Suppose further that* $J \subset \{1, \ldots, m\}$ *is a uniform random subset of size* ρm*, and that* $\boldsymbol{\sigma} \in \mathbb{R}^m$ *with entries of* $\boldsymbol{\sigma}_J$

i.i.d. ± 1 *(independent of* J*) and* $\boldsymbol{\sigma}_{J^C} = \mathbf{0}$*. Lastly assume that* m *is sufficiently large. Then with probability at least* $1 - C\exp(-\zeta^* m)$ *in* X*,* J*, and* $\boldsymbol{\sigma}$*, for all* $\boldsymbol{\alpha}_0$ *with* $\|\boldsymbol{\alpha}_0\|_0 \leq \beta^* m$ *and any* $\boldsymbol{z}_0$ *with sign vector* $\boldsymbol{\sigma}$ *and support* J*, we have*

$$(\boldsymbol{\alpha}_0, \boldsymbol{z}_0) = \arg\min_{\boldsymbol{\alpha}, \boldsymbol{z}} \|\boldsymbol{\alpha}\|_1 + \|\boldsymbol{z}\|_1 \text{ subject to } X\boldsymbol{\alpha} + \boldsymbol{z} = X\boldsymbol{\alpha}_0 + \boldsymbol{z}_0.$$

Here, C is a numerical constant and β^* and ζ^* are positive constants (independent of m) which depend on δ, ρ, and ν. By "ν sufficiently small" and "m sufficiently large," Wright and Ma mean that there exist constants $0 < \nu < \nu^*$ and $m > m^*$ (independent of m) such that $\nu^*(\delta, \rho) > 0$ and $m^*(\delta, \rho, \nu) > 0$, respectively.[1]

This theorem illustrates that ℓ^1/ℓ^0-equivalence can provably hold on highly-coherent data!

11.2. Project Description

Despite its applicability to highly-coherent data, Theorem 11.1.1 does not prove that ℓ^1/ℓ^0-equivalence holds in SRC. First of all, the theorem requires that m be sufficiently large, which may not be the case when dimensionality reduction is used. Second, the model in Theorem 11.1.1 does not explicitly deal with class-structured data. A true face recognition model should account for the individual subjects, with samples in the same class being (on average) more correlated than those from different classes. Thus our model should contain "sub-bouquets" (i.e., the classes) inside the larger bouquet.

With these changes in mind, we design an experiment to study the relationship between sparsity and ℓ^1-minimization on highly-coherent and class-structured data, such as the images used in face recognition. First, we specify the dimension m, the number of samples in each training class N_0, and the number of classes L. We require that $N_{\text{tr}} := N_0 L > m$ so that the resulting dictionary of training samples leads to an underdetermined system. We then randomly generate training data with an increasing amount of cone/bouquet structure as well as class structure, along with a test sample—with known sparse coefficient vector $\boldsymbol{\alpha}_0$—generated as a linear combination of training samples from a single class. We run a fixed number of trials of the experiment at each of 11

[1]The relationship between β^* and β is not explicitly stated, but it makes sense that $\beta^* \leq \beta$ by the proportional growth assumption. Further, if $\beta = \beta(\delta, \rho)$, then since $\beta^* = \beta^*(\delta, \rho, \nu)$, we can likely alternatively write $\beta^* = \beta^*(\beta, \nu)$.

increasing values of coherence (we call these *stages*) and determine at which stages ℓ^1-minimization can closely (or exactly) recover $\boldsymbol{\alpha}_0$.

11.3. Experiments

11.3.1. Experimental Setup. For each generated training set $X_{\text{tr}} = [X^{(1)}, \ldots, X^{(L)}] \in \mathbb{R}^{m \times N_{\text{tr}}}$, we set the (clean) test sample $\boldsymbol{y}_0$ to be a random vector in the positive span of the class 1 data. That is, we set

$$\boldsymbol{y}_0 := \alpha_1^{(1)} \boldsymbol{x}_1^{(1)} + \ldots + \alpha_{N_0}^{(1)} \boldsymbol{x}_{N_0}^{(1)},$$

where $X^{(1)} := [\boldsymbol{x}_1^{(1)}, \ldots, \boldsymbol{x}_{N_0}^{(1)}]$ and $\alpha_j^{(1)} \sim \text{unif}(0,1)$, $1 \leq j \leq N_0$. We then define

$$\boldsymbol{\alpha}_0 := [\alpha_1^{(1)}, \ldots, \alpha_{N_0}^{(1)}, 0, \ldots, 0]^{\mathsf{T}} \in \mathbb{R}^{N_{\text{tr}}}.$$

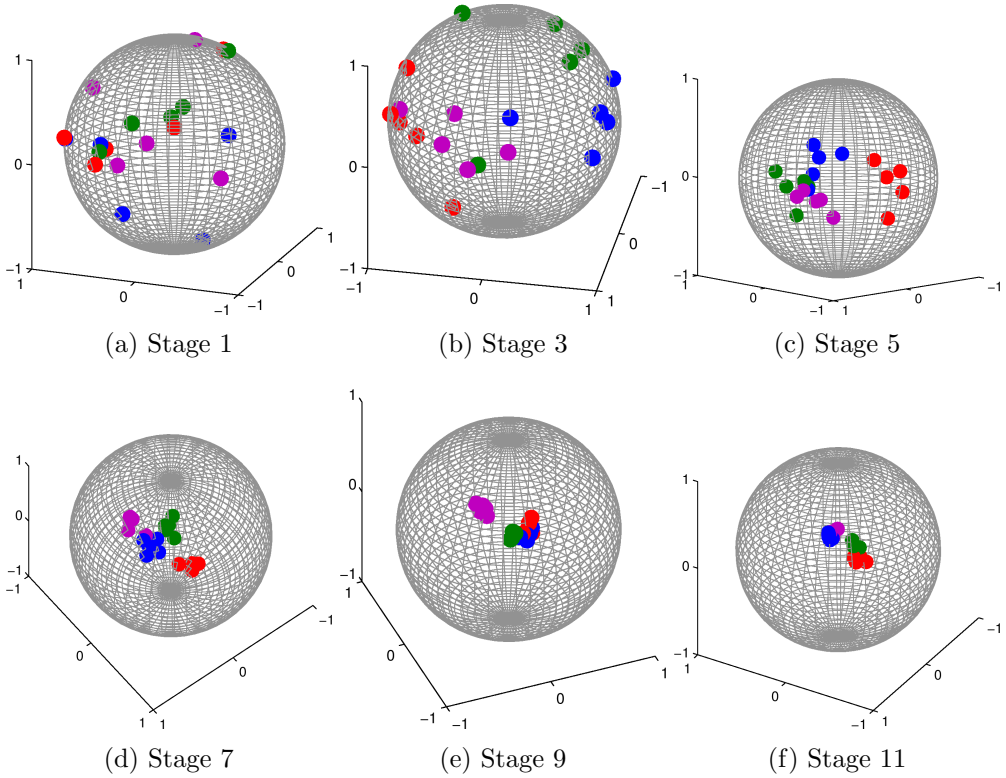
Given this setup, we want to see if ℓ^1-minimization will recover $\boldsymbol{\alpha}_0$, i.e., if the solution

$$\boldsymbol{\alpha}_1 := \arg \min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{tr}}}} \|\boldsymbol{\alpha}\|_1 \text{ subject to } X_{\text{tr}} \boldsymbol{\alpha} = \boldsymbol{y}_0$$

is equal to $\boldsymbol{\alpha}_0$. Note that for large L, $\boldsymbol{\alpha}_0$ can be viewed as a sparse vector.

In Stage 1 of our model, the training data has no class or cone structure and is randomly generated on the unit sphere S^{m-1}. It has been shown experimentally that, for $N_{\text{tr}} = 2m$ and m sufficiently large, an ℓ^1-minimization solution with no more than $(3/10)m$ nonzeros is enough to ensure it is the sparsest solution with high probability [**29**]. Thus we expect to see exact recovery in Stage 1 for values of N_0, m, and L satisfying these requirements.

To add both bouquet and class (or sub-bouquet) structure to the training set in subsequent stages, we define the cone mean $\overline{\boldsymbol{x}}$ and the class means $\{\overline{\boldsymbol{x}}_1, \ldots, \overline{\boldsymbol{x}}_L\}$. At Stage i, $1 \leq i \leq 11$, we set $\overline{\boldsymbol{x}} \sim \mathcal{N}(\boldsymbol{0}, I_m)$ and then modify $\overline{\boldsymbol{x}} \leftarrow \mu_i \overline{\boldsymbol{x}} / \|\overline{\boldsymbol{x}}\|_2$, where $\mu_i := (i-1)/10$ effectively increases the cone mean from 0 as i increases. Next, each class mean is randomly generated depending on $\overline{\boldsymbol{x}}$ as follows: For each class $1, \ldots, L$, we sample $\overline{\boldsymbol{x}}_l$ from $\mathcal{N}(\overline{\boldsymbol{x}}, \eta_i m^{-1/2} I_m)$ for $\eta_i := 2/i$ (so that each class mean becomes increasingly close to the cone mean) and then modify $\overline{\boldsymbol{x}}_l \leftarrow \mu_i \overline{\boldsymbol{x}}_l / \|\overline{\boldsymbol{x}}_l\|_2$, $1 \leq l \leq L$. Lastly, to generate the training samples in class $1 \leq l \leq L$, we sample $\boldsymbol{x}_j^{(l)}$ from



(a) Stage 1 (b) Stage 3 (c) Stage 5

(d) Stage 7 (e) Stage 9 (f) Stage 11

FIGURE 11.1. An example of the generated training data from the random database model across odd-numbered stages (as mutual coherence increases) with $m = 3$, $N_0 = 5$, and $L = 4$. The colors denote the classes. Plots have been manually rotated to aid in visualization. (a) At Stage 1, data is uniformly spread out on the sphere; (b)-(f) At increasingly higher stages, the data set as a whole becomes more bouquet-shaped, as does the data in each class.

$\mathcal{N}(\overline{\boldsymbol{x}}_l, (\eta_i m^{-1/2}/L) I_m)$ and then modify $\boldsymbol{x}_j^{(l)} \leftarrow \boldsymbol{x}_j^{(l)}/\|\boldsymbol{x}_j^{(l)}\|_2$, $1 \leq j \leq N_0$. Figure 11.1 shows an example of Stage $i \in \{1, 3, \ldots, 11\}$ with $m = 3$, $N_0 = 5$, and $L = 4$.

We perform experiments using four different specifications for the triples (N_0, m, L), as shown in Table 11.1. By design, we have that $\|\boldsymbol{\alpha}_0\|_0 = N_0$ in our experiments (though we will also briefly look at the case that $\|\boldsymbol{\alpha}_0\|_0 < N_0$). Note that (i) the inequality $\|\boldsymbol{\alpha}_0\|_0 < (3/10)m$ is satisfied for each of the specifications in Table 11.1 and (ii) these numbers are similar to what we might expect to see in classification of a face database (after some method of feature extraction is applied, as is generally required by SRC for face classification).

11.3.2. Experimental Results: No Noise.

ID	(N_0, m, L)	$\|\boldsymbol{\alpha}_0\|_0/m$	$\|\boldsymbol{\alpha}_0\|_0/N_{\text{tr}}$	Redundancy	Comments
DB-1	(5,50,20)	1/10	1/20	2:1	Baseline redundancy; N_0 small with respect to m, N_{tr}
DB-2	(10,50,10)	1/5	1/10	2:1	Baseline redundancy; N_0 less small with respect to m, N_{tr}
DB-3	(10,50,50)	1/5	1/50	10:1	Highly redundant; large L
DB-4	(5,200,50)	1/40	1/50	5:4	Barely redundant; large L

TABLE 11.1. Specification of parameters in the random database model.

Accuracy of recovery. We consider the following quantities for evaluating the success of ℓ^1/ℓ^0-recovery:

- The average normalized ℓ^2-error $\text{err}_{\ell^2} := \|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_0\|_2/\|\boldsymbol{\alpha}_0\|_2$ between the ℓ^1-minimized solution $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_0$,
- The average number of nonzeros of $\boldsymbol{\alpha}_1$ occurring at training samples *not* in class 1 (we call these "off-support" nonzeros, because they are nonzeros not in the support of $\boldsymbol{\alpha}_0$), divided by the total number of nonzeros. That is, let $\boldsymbol{\alpha}_1^{\text{off-supp}}$ be the result of setting all entries in $\boldsymbol{\alpha}_1$ that are in class 1 to zero. Then this error is defined as

$$\text{err}_{\text{supp}} := \frac{\|\boldsymbol{\alpha}_1^{\text{off-supp}}\|_0}{\|\boldsymbol{\alpha}_1\|_0},$$

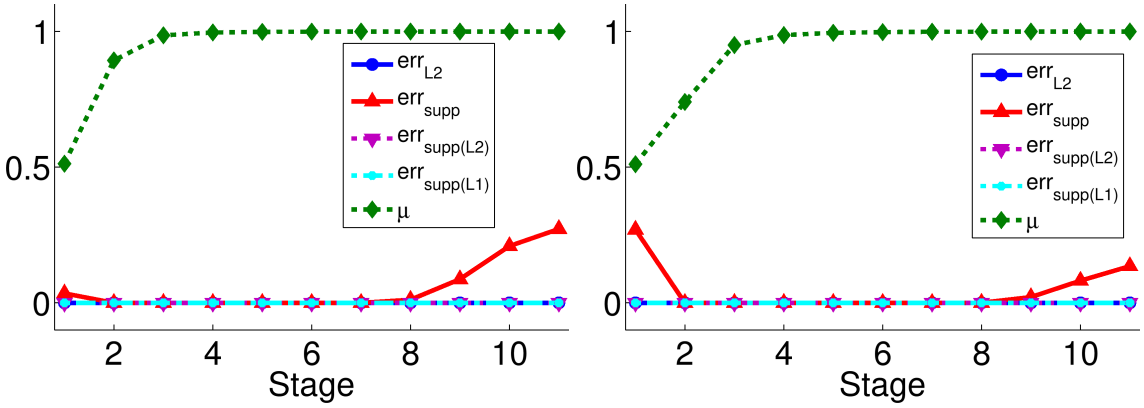
- Since err_{supp} does not provide information regarding the *size* of the off-support nonzero coefficients, we also consider

$$\text{err}_{\text{supp}(\ell^2)} := \frac{\|\boldsymbol{\alpha}_1^{\text{off-supp}}\|_2}{\|\boldsymbol{\alpha}_1\|_2} \text{ and } \text{err}_{\text{supp}(\ell^1)} := \frac{\|\boldsymbol{\alpha}_1^{\text{off-supp}}\|_1}{\|\boldsymbol{\alpha}_1\|_1},$$

- The average mutual coherence of the training set, $\mu(X_{\text{tr}}) =: \mu$.

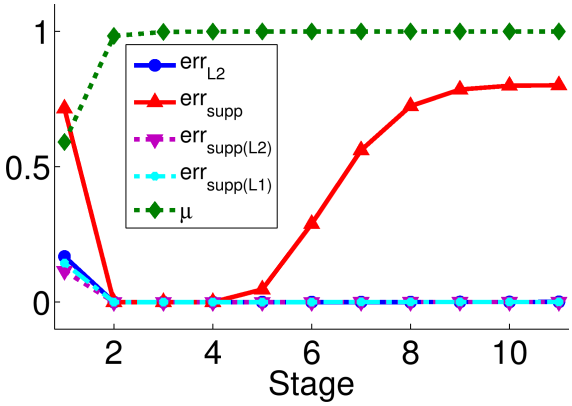
It is informative to consider the effect that the support error quantities would (hypothetically) have on the classification performance of SRC. Recall that SRC computes the class residuals $\text{err}_l(\boldsymbol{y}_0) := \|\boldsymbol{y}_0 - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}_1)\|_2$ (in the case that the clean test sample $\boldsymbol{y}_0$ is known) and assigns $\boldsymbol{y}_0$ to the class with the smallest residual. Thus if err_{supp}, $\text{err}_{\text{supp}(\ell^2)}$, and $\text{err}_{\text{supp}(\ell^1)}$ are small, we expect that SRC will have an easier time classifying the test sample correctly. For example, if all the support error quantities are 0, then $\delta_1(\boldsymbol{\alpha}_1) = \boldsymbol{\alpha}_1$ and it follows that the class 1 residual $\text{err}_1(\boldsymbol{y}_0) = 0$ and $\text{err}_l(\boldsymbol{y}_0) = \|\boldsymbol{y}_0\|_2$ for $2 \leq l \leq L$. This corresponds to the ideal classification scenario.

We compute the average quantities err_{ℓ^2}, err_{supp}, $\text{err}_{\text{supp}(\ell^2)}$, $\text{err}_{\text{supp}(\ell^1)}$, and μ over 1000 trials at each stage, using the ℓ^1-minimization algorithm HOMOTOPY with error/sparsity trade-off

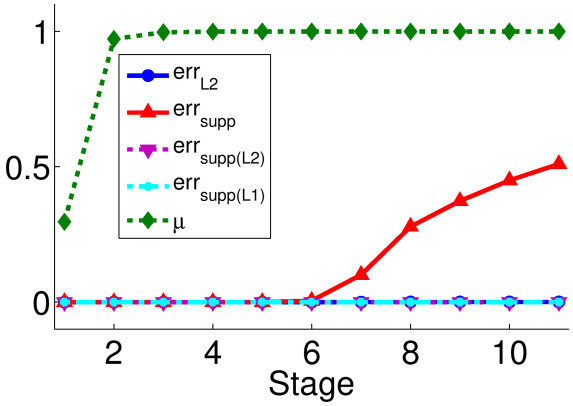


(a) DB-1: $(N_0, m, L) = (5, 50, 20)$

(b) DB-2: $(N_0, m, L) = (10, 50, 10)$



(c) DB-3: $(N_0, m, L) = (10, 50, 50)$



(d) DB-4: $(N_0, m, L) = (5, 200, 50)$

FIGURE 11.2. Recovery results on random database model (average of 1000 trials) in the case of no noise.

parameter $\lambda = 10^{-10}$ (to force near-exactness in the approximation). The results are shown in Figure 11.2.

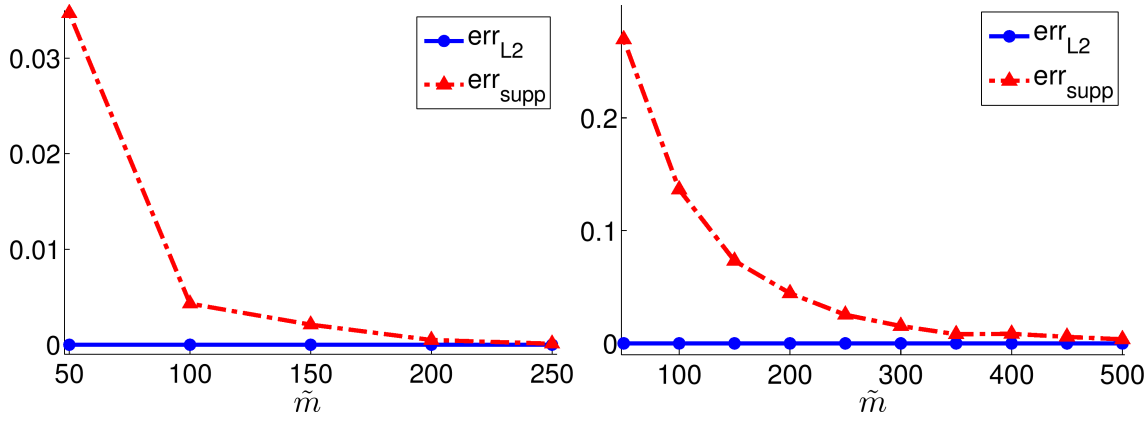
Considering that err_{supp} records *any* off-support nonzeros, regardless of how small, the results are quite good. In many cases, ℓ^1-minimization was able to recover the exact solution $\boldsymbol{\alpha}_0$ on highly-correlated data, and when errors in the support occurred, they were generally small.

We see two different things happening at either end of the Stage axis. At Stage 1, we see support errors in every database except DB-4 (the low-redundancy case). Further, there are nonzero values of err_{ℓ^2}, $\text{err}_{\text{supp}(\ell^2)}$, and $\text{err}_{\text{supp}(\ell^1)}$ for DB-3 (the high-redundancy case) at this stage. At high stages, we see similar small support errors as the data became very correlated; these support errors were numerous (accounting for around half the nonzero coefficients) for both DB-3 and DB-4.

We start by explaining the results at Stage 1. Given the plots in Figure 11.2, our instinct may be to suspect that something wrong happened here, especially considering the exact recovery on all databases at Stage 2. For the cases that we had a ratio of 2-to-1 redundancy, does this contradict the experimental result [**29**] that having $N_0 = \|\boldsymbol{\alpha}_0\|_0 < (3/10)m$ nonzeros guarantees ℓ^1/ℓ^0-equivalence with high probability? It would, but for the fact that this result holds asymptotically. To test this, we repeated the experiments for increasing values of m, scaling N_0 and L accordingly so that the redundancy remained constant. More precisely, we defined $r_1 := m/N_{\text{tr}}$ and $r_2 := N_0/L$ and then set $\tilde{L} := [\sqrt{\tilde{m}/(r_1 r_2)}]$ and $\tilde{N}_0 := r_2\tilde{L}$. Here, $[\,\cdot\,]$ denotes the nearest integer function and $\tilde{m}$, $\tilde{L}$, and $\tilde{N}_0$ denote the increased values of m, L, and N_0, respectively. As we illustrate in Figure 11.3, the value of err_{supp} decreased to 0 as $\tilde{m}$ increased. As is to be expected, both the amount of redundancy and the relationship N_0/m affected the speed of convergence. We exclude results for DB-4, as we already see perfect recovery at Stage 1 in Figure 11.2d.

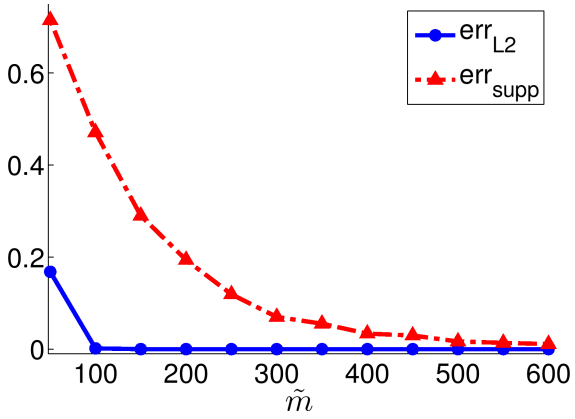
In comparing the Stage 1 results to those from data with bouquet/cone structure (i.e., Stages 2-11), it is initially surprising that small to moderate levels of correlation in the data samples appear to *improve* sparse recovery. As mentioned, we see near-perfect recovery of $\boldsymbol{\alpha}_0$ at Stage 2 for every tried (N_0, m, L) triple; this is in stark contrast to the recovery accuracy at Stage 1, especially for DB-3 (Figure 11.2c). This sharp change coincides with a significant increase in the within-class correlation between Stages 1 and 2 in our model, whereas the correlation between classes essentially remains unchanged. Though the exact specifics will depend on the ℓ^1-minimization algorithm used, we strongly suspect that the relative clustering of the samples in the support of $\boldsymbol{\alpha}_0$ at Stage 1 (as compared to their random distribution at Stage 1) make it much easier for the algorithm to recover the desired solution.

Conversely, at high stages, it appears that the loss of class structure negatively affected the recovery of $\boldsymbol{\alpha}_0$. As the standard deviation of the class mean distributions grew small, the class cones began to significantly overlap, and ℓ^1-minimization could not exactly recover the support of $\boldsymbol{\alpha}_0$. Notice that we see an especially large number of support errors err_{supp} for databases with large values of L, namely, DB-3 (Figure 11.2c) and DB-4 (Figure 11.2d). For DB-3, the nonzero values of err_{supp} at Stages 5 and 6 (compared to $\text{err}_{\text{supp}} \approx 0$ at these stages for DB-4) confirms that redundancy, as well as the number of classes, affects recovery.



(a) DB-1: $r_1 = 1/2$, $r_2 = 1/4$

(b) DB-2: $r_1 = 1/2$, $r_2 = 1$



(c) DB-3: $r_1 = 1/10$, $r_2 = 1/5$

FIGURE 11.3. Asymptotic recovery at Stage 1 of the random database model (average of 1000 trials). Note the different scales.

Effect on classification. We earlier discussed the relationship between the support error quantities $\mathrm{err}_{\mathrm{supp}}$, $\mathrm{err}_{\mathrm{supp}(\ell^2)}$, and $\mathrm{err}_{\mathrm{supp}(\ell^1)}$ on the classification performance of SRC, in particular, their effect on the class residuals $\mathrm{err}_l(\boldsymbol{y}_0) := \|\boldsymbol{y}_0 - X_{\mathrm{tr}}\delta_l(\boldsymbol{\alpha}_1)\|_2$. Here, we consider these residuals explicitly. For each of the four databases, we computed the average residual $\mathrm{err}_l(\boldsymbol{y}_0)$ (over 1000 trials) for each class $1 \leq l \leq L$ at each of the 11 values of coherence.

Not surprisingly given the small support error quantities determined in the previous section, there is a stark difference between the residual of class 1 and those of the other classes at all stages. More precisely, the ideal classification scenario occurs in all cases, with $\mathrm{err}_1(\boldsymbol{y}_0) \approx 0$ and $\mathrm{err}_l(\boldsymbol{y}_0) \approx \|\boldsymbol{y}_0\|_2$ for all $2 \leq l \leq L$. The approximations are of the order 10^{-8} (or better), except

for the highly-redundant database DB-3 at Stage 1. In this case, the average quantities were $\text{err}_1(\boldsymbol{y}_0) = 0.230$ and

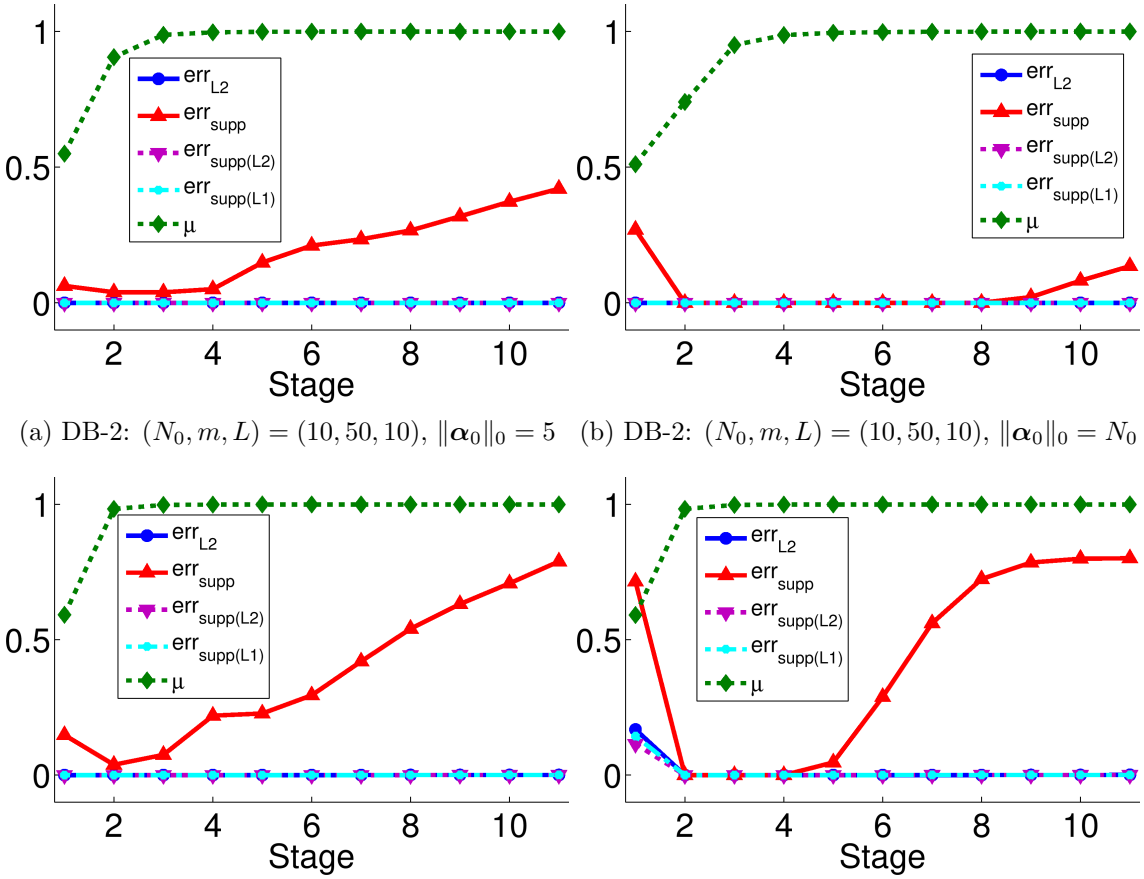
$$\|\boldsymbol{y}_0\|_2 - \underset{2\leq l\leq L}{\text{mean}}\, \text{err}_l(\boldsymbol{y}_0) = 0.004.$$

These findings are consistent with the results in Figure 11.2c. Even though these quantities at Stage 1 are nonzero, it is important to note that good classification would still be achieved, as $\min_{2\leq l\leq L} \text{err}_l(\boldsymbol{y}_0) = 1.806$, which is a lot larger than $\text{err}_1(\boldsymbol{y}_0) = 0.230$.

Varying the sparsity level. We next consider what happens when the sparsity level $\|\boldsymbol{\alpha}_0\|_0$ is strictly less than the number of class 1 training samples N_0. This is important to investigate: can ℓ^1-minimization identify the correct training samples from among the rest of the (highly-correlated) training data in that class? For DB-2 and DB-3, we generated $\boldsymbol{\alpha}_0$ (and subsequently $\boldsymbol{y}_0$) using the first five samples in class 1. Figures 11.4a and 11.4c show the recovery results, and Figures 11.4b and 11.4d repeat the plots in Figures 11.2b and 11.2c (in which $\|\boldsymbol{\alpha}_0\|_0 = N_0$) for convenient comparison.

At Stage 1, we see that the support of $\boldsymbol{\alpha}_1$ was more concentrated on the correct training samples when $\|\boldsymbol{\alpha}_0\|_0$ was smaller, evidenced by smaller values of err_{supp}. This is to be expected, as the ground truth solution became sparser. For the lower-redundancy case DB-2, we see far more support errors as the correlation increased when $\|\boldsymbol{\alpha}_0\|_0 = 5$ (Figure 11.4a) than for the case $\|\boldsymbol{\alpha}_0\|_0 = N_0$ (Figure 11.4b); however, the values of these off-support coefficients were very small, as demonstrated by the near-zero values of $\text{err}_{\text{supp}(\ell^2)}$ and $\text{err}_{\text{supp}(\ell^1)}$. Though class 1 training samples not in the support of $\boldsymbol{\alpha}_0$ were mistakenly selected as the data in class 1 became more correlated, these samples played a negligible role in the representation. For the high-redundancy case DB-3, we similarly see more small-valued, off-support coefficients at Stages 2-5 when $\|\boldsymbol{\alpha}_0\|_0 = 5$ (Figure 11.4c) than in the case $\|\boldsymbol{\alpha}_0\|_0 = N_0$ (Figure 11.4d). The value of err_{supp} for $\|\boldsymbol{\alpha}_0\|_0 < N_0$ was actually smaller than it was for $\|\boldsymbol{\alpha}_0\|_0 = N_0$ at many of the higher stages, however, suggesting that the added degree of sparsity helped to counter-balance the high redundancy of this database (and its negative effect on recovery) in these cases.

Eliminating errors by thresholding. Before we turn to the noisy setting, we demonstrate that the small support errors in $\boldsymbol{\alpha}_1$ depicted in Figure 11.2 can be completely remedied using thresholding in



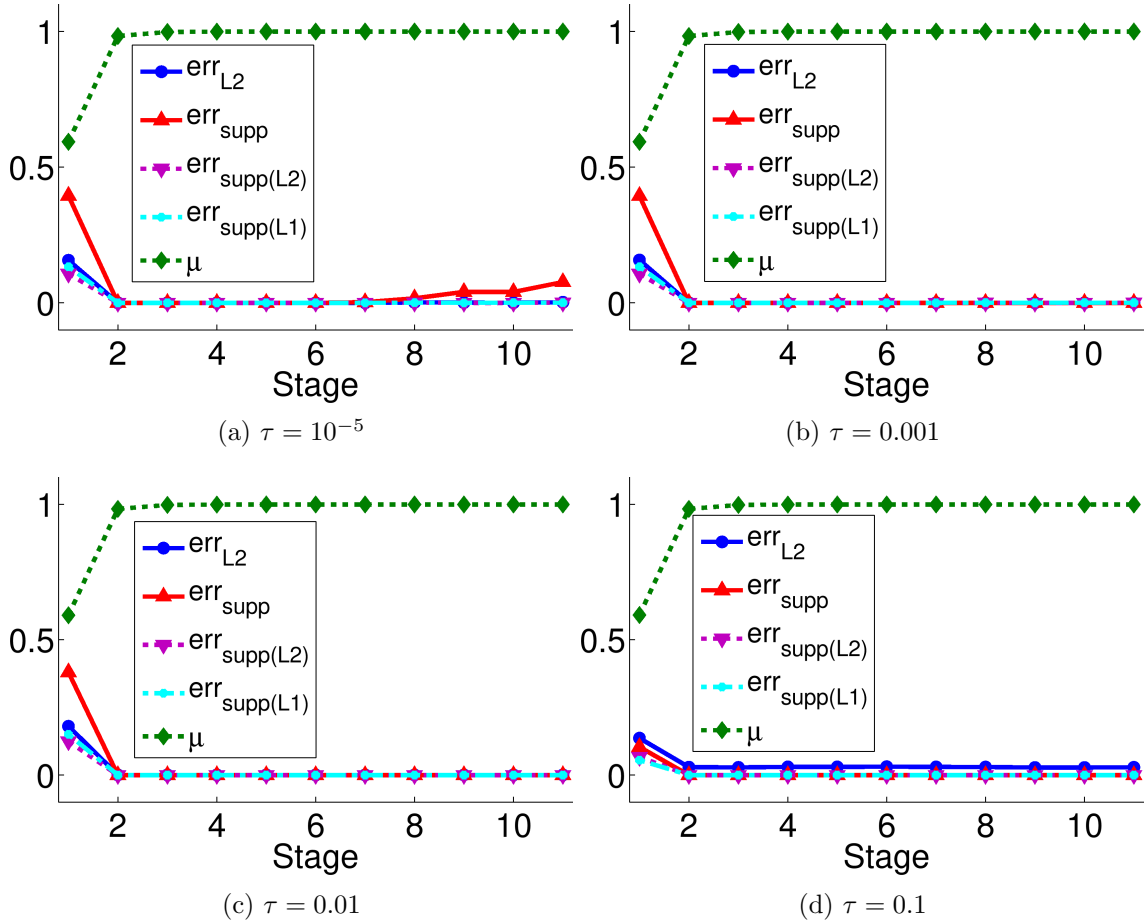
(a) DB-2: $(N_0, m, L) = (10, 50, 10)$, $\|\boldsymbol{\alpha}_0\|_0 = 5$ (b) DB-2: $(N_0, m, L) = (10, 50, 10)$, $\|\boldsymbol{\alpha}_0\|_0 = N_0$

(c) DB-3: $(N_0, m, L) = (10, 50, 50)$, $\|\boldsymbol{\alpha}_0\|_0 = 5$ (d) DB-3: $(N_0, m, L) = (10, 50, 50)$, $\|\boldsymbol{\alpha}_0\|_0 = N_0$

FIGURE 11.4. Comparing $\|\boldsymbol{\alpha}_0\|_0 < N_0$ and $\|\boldsymbol{\alpha}_0\|_0 = N_0$ sparsity levels (average of 1000 trials) on the random database model in the case of no noise.

all but the high-redundancy case DB-3. After determining $\boldsymbol{\alpha}_1$ as before, we set its small coefficients (those with absolute value less than some threshold τ) to zero, obtaining the vector $\boldsymbol{\alpha}_1^\tau$. We then re-solved the equation $X_{\text{tr}}\boldsymbol{\alpha} = \boldsymbol{y}_0$ with the constraint that the solution, denoted $\hat{\boldsymbol{\alpha}}_1$, had the same support as the thresholded $\boldsymbol{\alpha}_1^\tau$. For simplicity, we did this by setting the columns of X_{tr} corresponding to zero-coordinates in $\boldsymbol{\alpha}_1^\tau$ to $\mathbf{0}$, thus obtaining the matrix $\hat{X}_{\text{tr}}$. We then used MATLAB's "\" operator to define $\hat{\boldsymbol{\alpha}}_1 := \hat{X}_{\text{tr}} \backslash \boldsymbol{y}_0$. In our case, since X_{tr} was not square, the desired least squares solution was found by (MATLAB's implementation of) QR-factorization.

For all but the highly-redundant database DB-3, $\hat{\boldsymbol{\alpha}}_1$ was equal to the sparsest solution $\boldsymbol{\alpha}_0$ (up to nearly machine-precision) for the thresholding value $\tau = 10^{-5}$. For $\tau \in \{0.001, 0.01\}$ on these three databases (DB-1, DB-2, and DB-3), we saw small nonzero values of err_{ℓ^2}, but these errors



(a) $\tau = 10^{-5}$ (b) $\tau = 0.001$

(c) $\tau = 0.01$ (d) $\tau = 0.1$

FIGURE 11.5. The results of thresholding (average of 1000 trials) on the highly-redundant database DB-3: $(N_0, m, L) = (10, 50, 50)$ in the case of no noise.

were indiscernible in plots on the same scale as those in Figure 11.2, and so we do not show them here. For $\tau = 0.1$, there was a consistent, small but nontrivial ℓ^2-error across all stages, as small coefficients corresponding to class 1 training samples were incorrectly set to 0. For all four values of τ, there were no support errors.

For the high-redundancy case DB-3, we continued to see errors at Stage 1, similar to those in Figure 11.2c. For the thresholding values $\tau \in \{0.001, 0.01, 0.1\}$ (i.e., for τ large enough), there were no support errors at other stages. However, similarly to the other databases, we saw nontrivial ℓ^2-error when $\tau = 0.1$. We plot the results for DB-3 in Figure 11.5, stressing that the results for the other databases contained errors too small to produce nontrivial plots.

11.3.3. Experimental Results: Noisy Setting. In these experiments, we examine ℓ^1/ℓ^0-recovery when noise is added to the test sample $\boldsymbol{y}_0$. Recall the theorems by Donoho et al. regarding ℓ^1/ℓ^0-equivalence in the noisy setting stated in Theorems 9.2.5 and 9.2.6.

Accuracy of recovery. Unfortunately, Eq. (9.6) and Eq. (9.7) in the referenced theorems do not make sense for large mutual coherence $\mu(X)$. However, we can still look for a correlation between $\|\boldsymbol{\alpha}_0 - \boldsymbol{\alpha}_{1,\epsilon}\|_2$ (where $\boldsymbol{\alpha}_{1,\epsilon}$ is the solution to Eq. (11.4) below) and the values of the noise tolerance ζ (see that statement of Theorem 9.2.5), the approximation error bound ϵ, $N_0 = \|\boldsymbol{\alpha}_0\|_0 := k$, and $\mu(X_{\text{tr}})$, with $\epsilon =: C\zeta$ for some constant $C > 0$. We modify the experiments in Section 11.3.2 as follows: First, we specify the noise tolerance ζ and the constant C. After generating the training data and the (noise-free) test sample $\boldsymbol{y}_0$, we set $\boldsymbol{y} := \boldsymbol{y}_0 + \boldsymbol{z}$, where the entries of $\boldsymbol{z}$ are drawn from $\mathcal{N}(0, \zeta/(2\sqrt{m}))$. Then $\|\boldsymbol{z}\|_2 \leq \zeta$ with probability at least 95%. From here, we set $\epsilon := C\zeta$ and find

$$\boldsymbol{\alpha}_{1,\epsilon} := \arg\min_{\alpha} \|\boldsymbol{\alpha}\|_1 \text{ subject to } \|\boldsymbol{y} - X_{\text{tr}}\boldsymbol{\alpha}\|_2 \leq \epsilon. \tag{11.4}$$

We set $\zeta = 0.01$, and we used two values of C: $C = 5$, and $C = 10$, producing the (ζ, ϵ)-pairs $(0.01, 0.05)$ and $(0.01, 0.1)$. In order to ensure that the reconstruction error $\|X_{\text{tr}}\boldsymbol{\alpha}_{1,\epsilon} - \boldsymbol{y}\|_2$ was less than ϵ, we used the basis pursuit denoising version of the ℓ^1-minimization algorithm SPGL1 [**95**,**96**].

In Figure 11.6, we plot the normalized ℓ^2-error, the fraction of off-support nonzeros, the normalized ℓ^2 and ℓ^1-norms of the off-class support vectors, and the mutual coherence $\mu(X_{\text{tr}}) =: \mu$. Note that we modify the corresponding definitions given in Section 11.6 (for err_{ℓ^2}, err_{supp}, $\text{err}_{\text{supp}(\ell^2)}$ and $\text{err}_{\text{supp}(\ell^1)}$) to use $\boldsymbol{\alpha}_{1,\epsilon}$ instead of $\boldsymbol{\alpha}_1$ and do not change the notation. We report the averages over 1000 trials at each stage.

As we can see, there is clearly a relationship between err_{ℓ^2} and the amount of correlation in the data. As the data became increasingly bouquet-shaped, both within each class and as a data set as a whole, the normalized ℓ^2-distance between $\boldsymbol{\alpha}_{1,\epsilon}$ and $\boldsymbol{\alpha}_0$ increased. The rate of increase of this error appears to be related the redundancy of the database. It is evident that mutual coherence was not a good indicator of err_{ℓ^2}, as the plots show that err_{ℓ^2} could be relatively low even after $\mu(X_{\text{tr}})$ had reached its maximum value.

Perhaps more importantly, the supports of the solution vectors $\boldsymbol{\alpha}_{1,\epsilon}$ and $\boldsymbol{\alpha}_0$ were nearly identical at stages greater than 1. This means that the vast majority of nonzeros in $\boldsymbol{\alpha}_0$ occurred at positions

corresponding to class 1 training samples. To fix the small support errors, we could use the thresholding technique discussed in the previous section, choosing τ by trial-and-error. This method could also be used to ameliorate the numerous support errors for the databases DB-2 and DB-3 at Stage 1. In this case, we found that $\tau = 0.01$ greatly reduced the Stage 1 support errors but did not eliminate them completely.

Lastly, we consider the differences between setting $C = 5$ and $C = 10$. For the most part, the plots are quite similar. We see that setting $C = 5$ produced slightly better recovery than $C = 10$ at Stage 1, but in general, the normalized ℓ^2-error err_{ℓ^2} was the same for the two settings at higher stages. This is very informative, as it tells us that ℓ^1/ℓ^0-recovery on this kind of highly-correlated data is potentially quite robust to the setting of C in the approximation error tolerance $\epsilon = C\zeta$. Once again, we attribute this to the class structure of the data making it easier for the ℓ^1-minimization algorithm to find the class solution $\boldsymbol{\alpha}_0$.

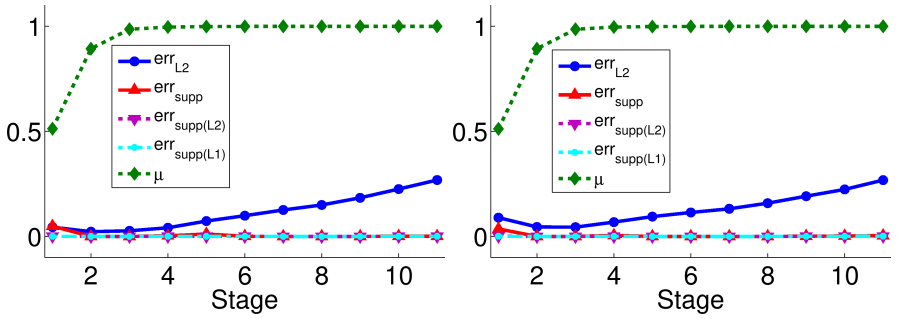
Effect on classification. As in the noise-free scenario, we compute the class residuals $\text{err}_l(\boldsymbol{y}) := \|\boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}_{1,\epsilon})\|_2$ for each of the four databases at each of the 11 values of coherence. Specifically, we are interested in how close the class 1 residual is to 0 (signifying perfect reconstruction of $\boldsymbol{y}$ using class 1) and how close the next smallest class residual $\min_{2\le l\le L}\text{err}_l(\boldsymbol{y})$ is to this value. If it is close, then it means that we should have less confidence in the SRC classification assignment than if these quantities were far apart, i.e., that SRC distinguishes the correct class less clearly.

The average relevant class residuals (over 1000 trials) are displayed in Table 11.2. Since the results for $C = 5$ and $C = 10$ were very similar, we only include the results for $C = 5$.

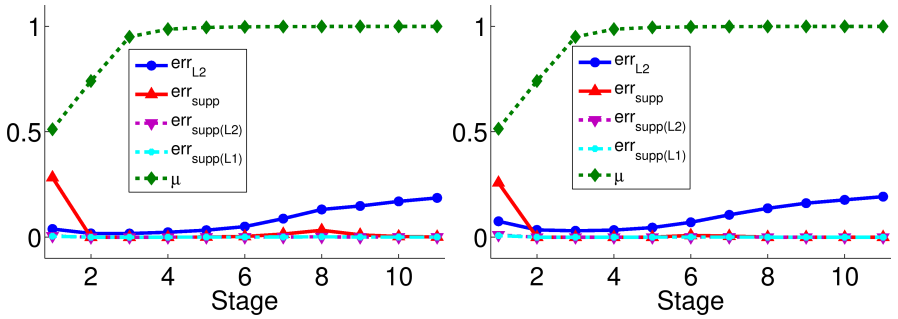
Noting that $\epsilon := C\zeta = 0.05$, we see that the ideal classification scenario occurred in nearly all cases. That is, since $\text{err}_1(\boldsymbol{y}) \approx \epsilon$ almost always, class 1 training samples made up essentially the entire approximation of the test sample. The exception, again, was DB-3 at Stage 1, for which $\text{err}_1(\boldsymbol{y})$ and $\min_{2\le l\le L}\text{err}_l(\boldsymbol{y})$ were the least separated (i.e., relatively close in value). However, correct classification would still be achieved.

The reader might notice that the quantities $\min_{2\le l\le L}\text{err}_l(\boldsymbol{y})$ at Stage 1 are lower than at higher stages; this is because

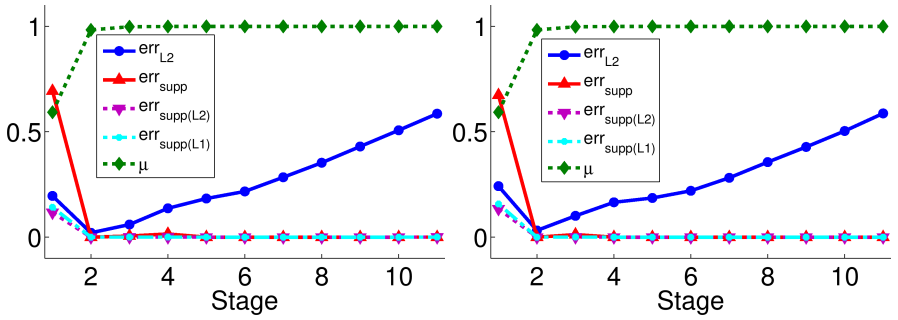
$$\min_{2\le l\le L}\text{err}_l(\boldsymbol{y}) = \min_{2\le l\le L}\|\boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}_{1,\epsilon})\|_2 \approx \|\boldsymbol{y} - X_{\text{tr}}\mathbf{0}\|_2 = \|\boldsymbol{y}\|_2$$



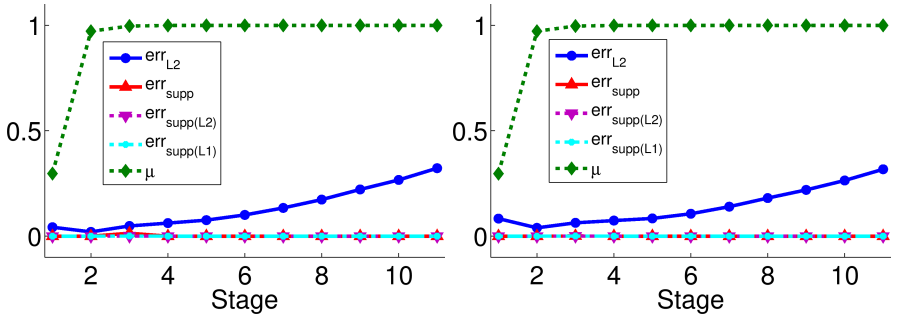
(a) DB-1, $C = 5$ (b) DB-1, $C = 10$



(c) DB-2, $C = 5$ (d) DB-2, $C = 10$



(e) DB-3, $C = 5$ (f) DB-3, $C = 10$



(g) DB-4, $C = 5$ (h) DB-4, $C = 10$

FIGURE 11.6. Recovery results on the random database model in the case of noise.

	DB-1		DB-2		DB-3		DB-4	
Stage	$\text{err}_1(\boldsymbol{y})$	$\min_{2\leq l\leq L} \text{err}_l(\boldsymbol{y})$	$\text{err}_1(\boldsymbol{y})$	$\min_{2\leq l\leq L} \text{err}_l(\boldsymbol{y})$	$\text{err}_1(\boldsymbol{y})$	$\min_{2\leq l\leq L} \text{err}_l(\boldsymbol{y})$	$\text{err}_1(\boldsymbol{y})$	$\min_{2\leq l\leq L} \text{err}_l(\boldsymbol{y})$
1	0.05	1.27	0.06	1.81	0.28	1.80	0.05	1.27
2	0.05	2.30	0.05	3.77	0.05	4.92	0.05	2.47
3	0.05	2.47	0.05	4.76	0.04	4.96	0.04	2.46
4	0.05	2.52	0.05	4.92	0.04	5.00	0.05	2.53
5	0.05	2.51	0.05	5.04	0.05	5.02	0.05	2.49
6	0.05	2.50	0.05	4.99	0.05	5.03	0.05	2.51
7	0.05	2.51	0.05	5.01	0.05	4.99	0.05	2.50
8	0.05	2.53	0.05	4.99	0.05	4.99	0.05	2.48
9	0.05	2.51	0.05	5.00	0.05	5.05	0.05	2.50
10	0.05	2.56	0.05	4.96	0.05	4.97	0.05	2.52
11	0.05	2.50	0.05	5.01	0.05	5.00	0.05	2.50

TABLE 11.2. Average SRC class residuals $\text{err}_1(\boldsymbol{y}) := \|\boldsymbol{y} - X_{\text{tr}}\delta_1(\boldsymbol{\alpha}_{1,\epsilon})\|_2$ and $\min_{2\leq l\leq L}\{\text{err}_l(\boldsymbol{y}) := \|\boldsymbol{y} - X_{\text{tr}}\delta_l(\boldsymbol{\alpha}_{1,\epsilon})\|_2\}$ (over 1000 trials) on the random database model in the case of noise.

is smaller in this case, due to the class 1 training samples being uniformly distributed on S^{m-1}.

11.4. Summary

In this project, we designed a model, inspired by the work of Wright and Ma [**102**], for facial recognition and other similar classification databases. To model the mechanisms of SRC [**104**], we randomly generated a test sample as a non-negative linear combination of a single class's training samples. We computed the corresponding (sparse) coefficient vector and then ran experiments to test whether or not ℓ^1-minimization, as it is used in the SRC setting, could recover this vector under increasing values of correlation, both within-class and in the database as a whole.

The results demonstrate that the within-class correlation in this model consistently improves ℓ^1/ℓ^0-recovery when compared to randomly-generated uniform data on the sphere. This is an important empirical result, as this latter type of data is one of the "golden children" of ℓ^1/ℓ^0-equivalence; i.e., these type of dictionaries produce, in some sense, ideal recovery (see, e.g., the work of Donoho [**29**]). However, those results are strongly asymptotic, and our experiments dealt only with small databases. More work is needed to determine if our findings hold up on larger data sets.

It is not too surprising, given the mutual coherence recovery condition studied in the last chapter, that very large correlation in the database as a whole can degrade recovery. When the

global correlation in our model was very high, so that the classes, or sub-bouquets, began to overlap, we saw that ℓ^1-minimization did not find the correct support of the sparse solution. However, we showed that the support could be completely fixed by a simple thresholding technique.

We also demonstrated that ℓ^1-minimization achieved a good approximation of the sparsest solution in the case of noise in our model. Though the accuracy of the approximation generally decreased as the data became more correlated, this deterioration was slow compared to the increase in mutual coherence of the database. Further, the amount of ℓ^2-error appeared to be less dependent on the relationship between noise ζ and error tolerance ϵ than it was on the amount of redundancy in the database.

Assuming that test samples truly are linear combinations of their ground truth class training samples, as is done in SRC, these experiments suggest that ℓ^1-minimization will recover this class representation, leading to good classification in SRC and similar classification algorithms. This of course assumes that our model is appropriate for the given data set, and that its values of N_0, m, and L are comparable to those used in our experiments, so that the class representation is sparse.

Our results are purely empirical; however, they strongly suggest that theoretical recovery results are possible. We conjecture that exact recovery can be provably obtained whenever the classes are sufficiently non-overlapping and that a similar result can be obtained in the case of noise. The amount of redundancy in the database and the number of classes will play a crucial role in this analysis.

Finally, though this project explicitly modeled the cone structure of facial images, our results are likely applicable to other areas of classification as well. In particular, as long as it is assumed that the training samples within each class are highly-correlated, we could amend our model so that the sign of each training sample was chosen randomly and so that the test sample was generated in the linear (not necessarily positive) span of its same-class training samples. However, since ℓ^1-minimization is invariant to multiplication of the dictionary elements by ± 1, we largely suspect that our results would be the same.

CHAPTER 12

Proving Equivalence in SRC via Nonlinear Embedding

12.1. This Approach

In this chapter, we consider *forcing* the mutual coherence criterion of ℓ^1/ℓ^0-equivalence to hold via data transformation. We want to analyze the effect of obtaining the *provably* sparsest solution in SRC on the algorithm's classification accuracy.

As we have seen in the previous two chapters, there is an innate conflict in using ℓ^1-minimization to obtain the sparsest solution in this context: we want the classes to be well-clustered, so that they may be easily classified, but if the data samples are not well-spread out, then we cannot prove that ℓ^1/ℓ^0-equivalence holds. As we have seen, class structure often results in large mutual coherence of the data set, making it impossible to apply either of the mutual coherence equivalence guarantees given in Theorem 9.2.4 and Theorem 9.2.5. We consider a resolution to this conflict through the use of *more space*. That is, if we had many "extra" dimensions, the data in each class could conceivably be spread out and we would still have enough "room" to keep the classes well-separated from each other.

Let us illustrate this in low dimension. Consider the toy example in which we have $L = 2$ classes, each containing $N_0 = 2$ samples in $\mathbb{R}^m$ for $m = 2$. First, let the goal be to arrange the samples in a way that minimizes their mutual coherence while at the same time provides some indication of class. Assuming that the samples must be normalized (as in SRC), this class-structure criterion can reasonably be interpreted as the requirement that

$$\left|\left\langle \boldsymbol{x}_i^{(1)}, \boldsymbol{x}_j^{(1)} \right\rangle\right| > \left|\left\langle \boldsymbol{x}_i^{(1)}, \boldsymbol{x}_j^{(2)} \right\rangle\right| \text{ and } \left|\left\langle \boldsymbol{x}_i^{(2)}, \boldsymbol{x}_j^{(2)} \right\rangle\right| > \left|\left\langle \boldsymbol{x}_i^{(2)}, \boldsymbol{x}_j^{(1)} \right\rangle\right|,$$

for $i, j \in \{1, 2\}$. In other words, the samples in the same class must be more correlated than samples in different classes.

One solution is given by the class matrices

$$X^{(1)} = \left[\boldsymbol{x}_1^{(1)}, \boldsymbol{x}_2^{(1)}\right] = \begin{bmatrix} 1 & \cos(\frac{\pi}{4} - \epsilon) \\ 0 & \sin(\frac{\pi}{4} - \epsilon) \end{bmatrix},$$

$$X^{(2)} = \left[\boldsymbol{x}_1^{(2)}, \boldsymbol{x}_2^{(2)}\right] = \begin{bmatrix} 0 & \cos(\frac{3\pi}{4} - \epsilon) \\ 1 & \sin(\frac{3\pi}{4} - \epsilon) \end{bmatrix}.$$

The (magnitude of) the dot product between samples in the same class is $\cos(\frac{\pi}{4} - \epsilon)$, and that of samples in different classes is $\cos(\frac{\pi}{4} + \epsilon)$. Clearly, the former quantity is the mutual coherence of the data set. This arrangement is illustrated in Figure 12.1a with $\epsilon = 0.2$.

Now, consider the same problem but in the case that we are given a third dimension. It is clear that we will be able to decrease the mutual coherence of the data set by moving samples into this extra space. One solution is given by the class matrices

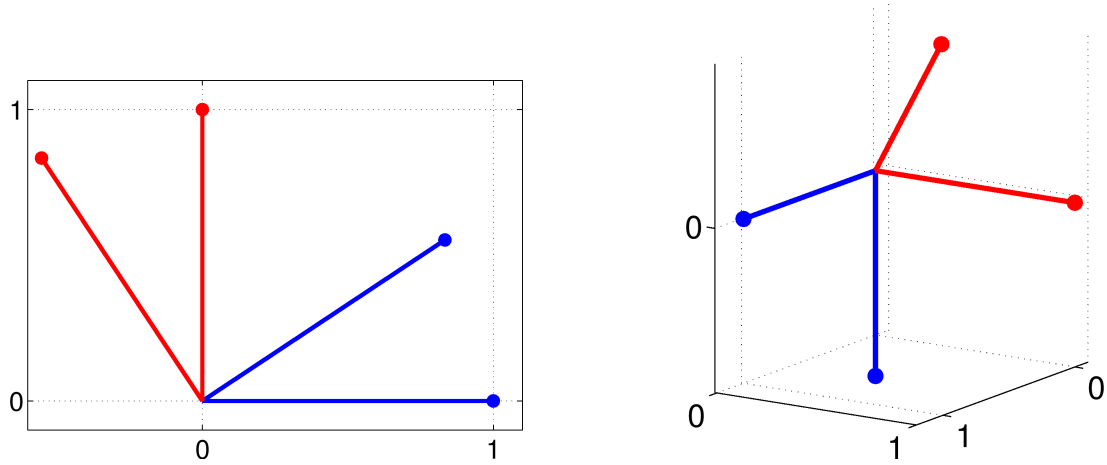
$$X^{(1)} = \begin{bmatrix} 1 & \cos(\theta_1)\sin(\phi_1) \\ 0 & \sin(\theta_1)\sin(\phi_1) \\ 0 & \cos(\phi_1) \end{bmatrix},$$

$$X^{(2)} = \begin{bmatrix} 0 & \cos(\theta_2)\sin(\phi_2) \\ 1 & \sin(\theta_2)\sin(\phi_2) \\ 0 & \cos(\phi_2) \end{bmatrix},$$

for $\theta_1 = \pi/4 - \epsilon$, $\theta_2 = \pi/4 + \epsilon$, $\phi_1 = 3\pi/4$, and $\phi_2 = \pi/4$. The mutual coherence of the data set is $\cos(\frac{\pi}{4} - \epsilon)\sin(\frac{3\pi}{4}) = \sin(\frac{\pi}{4} + \epsilon)\cos(\frac{\pi}{4})$. This arrangement is illustrated in Figure 12.1b with $\epsilon = 0.2$.

For $\epsilon = 0.2$, for example, adding an additional dimension allows us to decrease the mutual coherence of the data set from $\cos(\frac{\pi}{4} - \epsilon) \approx 0.8335$ to $\cos(\frac{\pi}{4} - \epsilon)\sin(\frac{3\pi}{4}) \approx 0.5894$. This is a substantial decrease.

The reader may question the practical relevance of considering such a transformation. We admit that our motivation behind this project is mostly theoretical: to what extent does finding the sparsest solution in SRC affect classification? Is the reward of true sparsity worthwhile, or is it enough in the SRC framework to achieve merely approximately-sparse solutions? Without knowing



(a) $m = 2$, $\mu = 0.8335$ (b) $m = 3$, $\mu = 0.5894$

FIGURE 12.1. Illustration of decreasing the mutual coherence of a data set by embedding it into higher dimension. (a) Samples in original space $\mathbb{R}^2$, (b) Samples in transform space $\mathbb{R}^3$. Colors denote classes.

the sparsest solution, these questions cannot be answered. Rather than using a combinatorial solver to determine the sparsest solution $\boldsymbol{\alpha}_0$ (an impractical undertaking for large data sets), we consider a transformation of the data that allows us to apply Theorem 9.2.4.

12.2. Designing the Transform

12.2.1. Considering an Explicit Transform. Ideally, we seek a transform $\phi^* : \mathbb{R}^m \to \mathbb{R}^{\tilde{m}}$ with $m < \tilde{m}$ that satisfies

$$\phi^* = \arg\max_{\phi \in \mathcal{C}} f_{\text{cs}}([\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_{N_{\text{tr}}})]) \text{ subject to } \mu([\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_{N_{\text{tr}}})]) \leq \tilde{\mu},$$

where $\mathcal{C}$ is some compact set (so that f_{cs} obtains a maximum). Here, f_{cs} evaluates the amount of class structure in the transformed database ("cs" stands for "class structure"), and $\tilde{\mu}$ is an upper bound on the transformed database's mutual coherence. For example, f_{cs} might denote the inverse of the sum of within-class distances or the inverse of the Frobenius norm of the within-class scatter matrix used in *linear discriminant analysis* [**42**, **74**].

We note that ϕ^* must be a nonlinear transform, otherwise the dimension of the subspace containing the embedded samples will be no greater than that of the original space (m). Thus we

will have failed to utilize the extra space (needed to achieve our objective) awarded by the increased ambient dimension $\tilde{m}$.

12.2.2. Using the Kernel Method. Rather than constructing an explicit transform, we consider the reduction of mutual coherence via the so-called "kernel trick." Essentially, the kernel trick allows us to perform operations in a space of dimension $\tilde{m} > m$ (possibly infinite-dimensional) without having to actually compute the transformed samples. The "trick" is to work only with the inner-products between transformed samples, which are given to us by some kernel function $\kappa : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$.

More formally, denote the transform by ϕ. We will write $\Phi(X_{\text{tr}}) := [\phi(\boldsymbol{x}_1), \ldots, \phi(\boldsymbol{x}_{N_{\text{tr}}})]$ to denote the matrix of training samples in the transform (or *kernel*) space. We define the inner-product in the kernel space as

$$\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle =: \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

for $1 \leq i, j \leq N_{\text{tr}}$. The kernel function κ should satisfy *Mercer's condition*[1] so that κ defines a proper inner-product [**22**].

Kernel methods can be particularly effective when used to "non-linearize" linear classifiers. In *kernel support vector machines*, for example, classes that are not linearly-separable in the original space may be separated linearly in kernel space (see the work of Boser et al. [**7**]). Though SRC is not linear, it does assume a linear relationship between the test sample and the training samples in its ground truth class. When such a relationship does not hold in the original space, it may hold in kernel space given that an appropriate kernel is selected [**111**].

The kernel trick may seem like magic, but it has a drawback. Namely, the linear structure that we desire in kernel space cannot be attained by just any kernel. Kernel methods require some prior knowledge about the data and the classification task in order to select a kernel that will lead to the desired structure in kernel space.

12.2.3. The Gaussian Kernel. We consider the *Gaussian* kernel, given by

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) := e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}}.$$

[1]The kernel κ satisfies Mercer's condition if $\iint \kappa(\boldsymbol{x}, \boldsymbol{y}) g(\boldsymbol{x}) g(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y} \geq 0$ for all square-integrable functions g.

Essentially, the Gaussian kernel adds *inverse exponential scaling* to the Euclidean distance function. Points close together obtain values of κ that are close to 1, whereas points that are faraway from each other have kernel values approaching 0. The *window* or *width* parameter σ controls the drop off (or steepness) of this trade-off.

Recall that our goal in the transform space is to decrease mutual coherence while still maintaining some degree of class structure in the data. In kernel space, the mutual coherence of the (transformed) training set is given by

$$\begin{aligned}\mu(\Phi(X_{\text{tr}})) &= \max_{1\leq i\neq j\leq N_{\text{tr}}} |\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle| \\ &= \max_{1\leq i\neq j\leq N_{\text{tr}}} |\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)| \\ &= \max_{1\leq i\neq j\leq N_{\text{tr}}} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2}{\sigma^2}}.\end{aligned}$$

Since the vectors $\hat{\boldsymbol{x}}_i$ and $\hat{\boldsymbol{x}}_j$ satisfying $\|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j\|_2 = \max_{i\neq j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$ are fixed for a given training set, the mutual coherence $\mu(\Phi(X_{\text{tr}}))$ depends completely on σ. Thus we can write $\mu(\Phi(X_{\text{tr}})) = \mu = \mu(\sigma)$. To reiterate, we can completely control the mutual coherence of the data in the kernel space by adjusting σ. Thus the Gaussian kernel appears to be a good choice for use in our task, assuming that we also have some means of controlling the amount of class structure in the data set in kernel space.

How do we help ensure that class structure is maintained as the mutual coherence decreases? As $\sigma \to 0$, $\mu(\sigma) \to 0$. In kernel space, this means that the training samples become closer to being pairwise orthogonal, the result of the inner-products between samples in kernel space—equivalently, their Gaussian kernel values in the original space—going to 0. Although the classes eventually become indistinguishable as $\sigma \to 0$, samples that are close together (with respect to Euclidean distance) in the original space take longer to become pairwise orthogonal than samples that are far apart. It follows that data sets that will maintain their class structure better for small values of σ are exactly those that are well-separated (in the original space) by Euclidean distance. Thus the class structure criterion can be satisfied by choosing a database with this property.

This setup might raise some objections. In the first place, were our aim simply to classify a given data set, this line of approach would be circular. If we knew the data were easily classified

using Euclidean methods, we could simply apply an algorithm such as 1NN in the original space. Secondly, often data sets cannot be well-separated using Euclidean distance, especially when the data from each class is drawn from a nonlinear manifold.

However, our aim in this experiment is *not* to classify an arbitrary data set or to validate the efficacy of the kernel method for use in classification. As we expand upon in the next section, our goal is to *leverage the kernel setup* in order to investigate the relationship between the mutual coherence ℓ^1/ℓ^0-equivalence guarantee, the sparsity of the coefficient vector, and the classification accuracy of SRC. It is a tool to access the *provably sparsest solution*, something that we cannot (tractably) do in general in the original space. Further, remember that the key to successfully using kernel methods is to select a kernel designed to produce the desired structure in kernel space. For our purposes, this is the *preservation of class structure as the samples in kernel space become less correlated*. When the Gaussian kernel is used, this exactly means that Euclidean distance is a good indicator of class in the original space.

12.2.4. The Goal. Our goal is to use the kernel trick with the Gaussian kernel to achieve ℓ^1/ℓ^0-equivalence in kernel space. We want to investigate the effect of this (provable) equivalence on the classification accuracy of SRC.

In order to ensure ℓ^1/ℓ^0-equivalence, we will choose the Gaussian width parameter σ so that the mutual coherence is small enough that Theorem 9.2.4 holds. Let us set

$$k_{\text{sup}} := \frac{1}{2}\Big(1 + \frac{1}{\mu}\Big).$$

Clearly, k_{sup} completely depends on μ, or equivalently, on σ. As σ approaches 0, $k_{\text{sup}} = k_{\text{sup}}(\sigma)$ blows up. Since class structure deteriorates as $\sigma \to 0$ (as samples become more and more pairwise orthogonal), we want to choose σ to be the largest value such that, with high probability (whp), the sparsity level $\|\boldsymbol{\alpha}_1\|_0$ is less than k_{sup}, where $\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}^* \in \mathbb{R}^{N_{\text{tr}}}$ is the solution to the exact ℓ^1-minimization problem in SRC given by Eq. (3.5) (replacing X_{tr} with $\Phi(X_{\text{tr}})$ and $\boldsymbol{y}$ with $\phi(\boldsymbol{y})$). This will ensure that $\boldsymbol{\alpha}_1$ is the sparsest solution by Theorem 9.2.4. Using "mc" to denote "mutual coherence," we define

$$\sigma_{\text{mc}} := \max\Big\{\sigma : \|\boldsymbol{\alpha}_1\|_0 \overset{\text{whp}}{<} k_{\text{sup}}\Big\}. \tag{12.1}$$

Consider now what happens as $\sigma \to \infty$. We have that $\mu \to 1$ and $k_{\sup}(\sigma) \to 1$. More informatively, the samples in the kernel space become increasingly correlated, and class structure is eventually lost. Another way of putting this is that as $\sigma \to 0$, "within-class" structure is lost, and as $\sigma \to \infty$, "between-class" structure is lost. In both cases, the data becomes harder to classify using SRC-based classifiers. It follows, as for all kernel classifiers when the Gaussian kernel is used, that there is a set of values of σ (possibly containing only one point) that produces a sufficiently high degree of class structure so that maximum classification accuracy is achieved for all values in this set (whp). We denote the maximum value in this set by σ_{acc}.

We want to investigate the relationship between σ_{mc} and σ_{acc}. We are also interested in the sparsity level $\|\boldsymbol{\alpha}_1\|_0$ at both $\sigma = \sigma_{\text{mc}}$ and $\sigma = \sigma_{\text{acc}}$. Since some coefficients may be small, we lastly should consider the size of the coefficients of training samples corresponding to the ground truth class of $\boldsymbol{y}$. In analyzing these quantities and relationships, we aim to provide insight into the role of sparsity in classification.

12.3. An Unfortunate yet Necessary Modification

On a toy database, we confirmed that the kernel method can work exactly as we want it to, producing training samples in kernel space that have small enough within-class correlation so that the bound in Eq. (9.4) holds, while still distinguishing classes via even smaller between-class coherence. For very well-separated data (in the original space), we found that when $\sigma = \sigma_{\text{mc}}$, the coherence between training samples in different classes, and that between the test sample and training samples not in its ground truth class, were actually zero (in kernel space). It seems that this should allow for very good classification by SRC in kernel space *and* provable ℓ^1/ℓ^0-equivalence of the solution vector, allowing us to investigate the change in classification accuracy as σ increases and the solution becomes denser.

Unfortunately, as $\sigma \to 0$, we lose the ability to write $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$ for *any* coefficient vector $\boldsymbol{\alpha}$. Recall that this equality is a key aspect of the mutual coherence condition in Theorem 9.2.4. In decreasing σ, we cause not only the training samples to become more orthogonal to each other, but also the test sample to become more orthogonal to each training sample, to the point that when

$\sigma = \sigma_{\text{mc}}$, $\phi(\boldsymbol{y})$ is likely not contained in the span of the columns of $\Phi(X_{\text{tr}})$. We state this formally in the following proposition:

PROPOSITION 12.3.1. *Given the kernel setup outlined above, suppose that we take σ small enough so that the mutual coherence bound $\|\boldsymbol{\alpha}\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(\Phi(X_{\text{tr}}))}\right)$ holds, i.e., $\sigma \leq \sigma_{\text{mc}}$. Then*

$$\Phi(X_{\text{tr}})\boldsymbol{\alpha} \neq \phi(\boldsymbol{y})$$

with high probability, for all $\boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{tr}}}$.

Proof: This is a direct consequence of Corollary 10.4.3. □

To summarize, a consequence of our kernel space setup is the problem of "too much space:" the resulting system is overdetermined with no solution to $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$ when $\sigma \leq \sigma_{\text{mc}}$. By Theorem 9.2.4, the minimal ℓ^1-norm solution satisfying $\Phi(X_{\text{tr}})\boldsymbol{\alpha}_1 = \phi(\boldsymbol{y})$ with $\|\boldsymbol{\alpha}_1\|_0 < (1/2)(1 + (1/\mu))$ is necessarily the sparsest such solution. However, if there is no solution satisfying $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$, then there can be no sparsest solution!

Even when the equality in SRC is relaxed and the constrained ℓ^1-minimization problem in Eq. (9.3) is used,[2] relating the found solution $\boldsymbol{\alpha}_{1,\epsilon}$ and the true sparsest solution $\boldsymbol{\alpha}_0$ using Theorem 9.2.5 requires the *existence* of some $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0$ satisfying the equality $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$. Since the bound in Eq. (9.5) in the noisy case is more restrictive than Eq. (9.4) in the noiseless case, to satisfy Theorem 9.2.5 we must have $\sigma < \sigma_{\text{mc}}$. By Proposition 12.3.1, no such $\boldsymbol{\alpha}$ exists, and it follows that Theorem 9.2.5 cannot be applied in this setup, either.

We can avoid this debilitating problem by generating $\phi(\boldsymbol{y})$ as a linear combination of the columns of $\Phi(X_{\text{tr}})$ (in particular, with nonzero coefficients occurring at training samples in the ground truth class of $\boldsymbol{y}$). In this manner, we can ensure that $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$ has a solution for all values of σ. However, this will mean that we never actually compute or handle the test sample $\boldsymbol{y}$ in the original space and only *assume that it exists implicitly*, working solely with the inner-products between samples in kernel space. This is a little weird! The reader may object that we cannot just make up test samples in this manner, and in general, this is absolutely true. Nevertheless, we

[2]Note that the formulation in Eq. (9.3) is equivalent to the regularized ℓ^1-minimization problem in Eq. (3.7) in the formal SRC algorithm statement.

reiterate that our goal in this experiment is not to validate the kernel approach or to classify an arbitrary database, and so the implied existence of $\boldsymbol{y}$ is acceptable in this context.

We admit, though, that this modification is not ideal, and it changes the experiment significantly. In fact, our whole motivation must change, as we no longer need to use Theorem 9.2.4 to find a sparse solution. For all values of σ, we now know that there exists a solution $\boldsymbol{\alpha}$ to $\Phi(X_{\text{tr}})\boldsymbol{\alpha} = \phi(\boldsymbol{y})$ having no more than $\|\boldsymbol{\alpha}\|_0 = N_l$ nonzero coefficients, where $\boldsymbol{y}$ is in class l.

Additionally, as $\sigma \to 0$ and the training data become closer to orthogonal, we will never lose the relationship $\phi(\boldsymbol{y}) \in \text{span}\{\phi(\boldsymbol{x}_1^{(l)}), \ldots, \phi(\boldsymbol{x}_{N_l}^{(l)})\}$. Thus we will not see the classification performance deteriorate *at all* as $\sigma \to 0$. In other words, decreasing μ so that we can provably obtain ℓ^1/ℓ^0-equivalence in this setup can only *help* classification accuracy, as doing so isolates the linear relationship between the test sample and the training samples in its ground truth class (in kernel space). As a consequence, we do not need to explicitly require that the database be well-separated by Euclidean distance in the original space, since this imposed linear relationship will essentially do this for us. We stress that this is certainly not the case in general: consider the increasing difficulty of identifying class structure in a data set whose samples become more and more uncorrelated. Even if the classes are very well-separated in the original space, there will be a certain point at which all samples become orthogonal and class structure is entirely lost. Thus generating $\phi(\boldsymbol{y})$ in this manner adds an undesirable degree of artificiality into our experiment.

However, we will show that we can still gain informative insights from observing the relationship between σ_{mc} and σ_{acc} in this setup. Further, we will observe the sparsity level of the coefficient vectors at σ_{acc} and investigate how σ_{acc} depends on the difficulty of the classification task. Note that these goals make this experiment fundamentally different from those in Chapter 11, wherein we primarily focused on the recovery of the sparsest solution using ℓ^1-minimization and never actually performed classification using SRC. We also did not consider the relationship between classification accuracy and the mutual coherence bound in Eq. (9.4) in that project. Thus our work here will provide new insights.

12.4. Experiments

12.4.1. Experimental Setup. For a fixed training set and fixed σ, we generate N_l test samples in kernel space for each class $1 \leq l \leq L$ as linear combinations of the training samples in that class (in kernel space) with coefficients randomly drawn from unif$(0, 1)$ distribution. Non-negative coefficients are used so that $\langle \phi(\boldsymbol{y}), \phi(\boldsymbol{x}_j) \rangle \geq 0$ for $1 \leq j \leq N_{\text{tr}}$, as is consistent with the Gaussian kernel. We then apply SRC in kernel space to classify the resulting test samples, using the Kernel SRC algorithm of Kang et al., in particular, their *kernel coordinate descent* (KCD) algorithm [**55**]. Note that in their paper, the authors apply this algorithm to the *local binary patterns* of the original samples instead of the original samples themselves, and since other types of kernels are more appropriate for these type of features, they do not use the Gaussian kernel, as we do.

In our experiments, we determine σ_{mc} and σ_{acc} by trial-and-error. Given the randomness inherent in the database construction (see Section 12.4.3 for a description of the database used), we request that the reader grant us leeway to make judicious choices in terms of rounding, etc., in determining these values. Additionally, note that we thresholded the entries of each ℓ^1-minimized coefficient vector $\boldsymbol{\alpha}_1$ by 10^{-10} to standardize machine precision.

12.4.2. An Upper Bound. We saw in Chapter 10 that we cannot apply Theorem 9.2.4 unless $\mu < \frac{1}{3}$. Since we are using the kernel approach, this means that we must have

$$\mu(\Phi(X_{\text{tr}})) = \max_{1 \leq i \neq j \leq N_{\text{tr}}} \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle = \max_{1 \leq i \neq j \leq N_{\text{tr}}} \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) < \frac{1}{3}.$$

In particular, since we are using the Gaussian kernel, it must be the case that

$$\begin{aligned} \max_{i \neq j} \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \max_{1 \leq i \neq j \leq N_{\text{tr}}} e^{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}} &< \frac{1}{3} \\ \Rightarrow \sigma < \frac{1}{\sqrt{\log(3)}} \max_{1 \leq i \neq j \leq N_{\text{tr}}} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2. \end{aligned}$$

Since the training samples (in the original space) are normalized, this means that

$$\sigma < \frac{2}{\sqrt{\log(3)}} \approx 1.35. \tag{12.2}$$

Thus in searching for σ_{mc}, we only need to consider values of σ less than 1.35.

12.4.3. Database Description. We constructed a very simple toy database in the original space as follows: Samples in the lth class were initially N_0 copies of the canonical basis vector $\boldsymbol{e}_l \in \mathbb{R}^L$, where L was the number of classes. The feature dimension m was user-specified, and then $m - L$ coordinates were added to each canonical basis vector and set to zero. Lastly, random noise from $\mathcal{N}(0, \eta^2)$ was added to all (training) samples in all coordinates.

We set $N_0 = 5$, $m = 50$, and $L = 20$, so that each class would consist of a relatively small portion of the dictionary X_{tr}, as is ideal in SRC. We used three different values of noise level $\eta \in \{0.001, 0.1, 0.5\}$. As in the ℓ^1-minimization algorithm HOMOTOPY, KCD requires an error/sparsity tradeoff parameter λ. To force near-exactness in the representations, we set $\lambda = 10^{-10}$.

Note that one advantage of the kernel setup in this context is the ability to apply it to *any* training database. This is another critical difference between this project and the experiments in Chapter 11, which were fundamentally based around a particular database model.

REMARK 12.4.1. *The reader may question why, in this project, we did not use the synthetic database from the experiments in Chapters 5 and 7, or why we used a different synthetic database than the one in the last chapter, so that we might obtain a* fair comparison*. We are making our best effort to stress that this line of thinking misconstrues the point of this experiment. Here, we only care about the classification results of Kernel SRC as they relate to the sparsity level $\|\boldsymbol{\alpha}_1\|_0$ and the mutual coherence bound in Eq.* (9.4)*. We are completely uninterested in whether the kernel approach improves the classification accuracy of SRC (for a positive answer to this question, see, for example, Kang et al.'s paper [**55**]). Further, the aforementioned synthetic databases were designed for specific purposes: the sinusoidal wave database used in the LPCA-SRC experiments was constructed so that LPCA-SRC and SRC could be compared in the case of nonlinear and intersecting class manifolds, and the database model in Chapter 11 was chosen so that ℓ^1/ℓ^0-equivalence—and not, specifically, classification performance in SRC—could be studied at increasing levels of data correlation. So that the aims of these previous experiments did not bleed into our goals here, we used a completely new (and very simple) database.*

12.4.4. Results. In Figure 12.2, we plot the synthetic database results for each value of η over various values of σ, annotating the values of σ_{mc} and σ_{acc}. We report the averages over 100

instantiations of the training and test sets ("trials"). In particular, we report the average sparsity level, Kernel SRC classification accuracy, and the (relative) ℓ^2 and ℓ^1-norms of the correct class support. These quantities are defined rigorously as

$$\text{Sparsity} := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{\text{all test samples}}{\text{median}} \frac{\|\boldsymbol{\alpha}_1\|_0}{N_{\text{tr}}} \right\}$$

for $\boldsymbol{\alpha}_1$ thresholded at 10^{-10} (we compute the median sparsity over all test samples so that the result is more robust to atypical very sparse or very dense coefficient vectors),

$$\text{Accuracy} := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{\text{all test samples}}{\text{mean}} \mathbb{1}_{\{\text{class\_label}\,(\boldsymbol{y})=\text{ground\_truth\_class}(\boldsymbol{y})\}} \right\}$$

where $\mathbb{1}_{\{x=y\}}$ is the indicator function that returns 1 if $x = y$ and 0 otherwise, and

$$\text{supp}(\ell^2) := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{\text{all test samples}}{\text{mean}} \frac{\|\delta_{\text{GT}}(\boldsymbol{\alpha}_1)\|_2}{\|\boldsymbol{\alpha}_1\|_2} \right\}, \quad \text{supp}(\ell^1) := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{\text{all test samples}}{\text{mean}} \frac{\|\delta_{\text{GT}}(\boldsymbol{\alpha}_1)\|_1}{\|\boldsymbol{\alpha}_1\|_1} \right\},$$

where the nonzero entries of $\delta_{\text{GT}}(\boldsymbol{\alpha}_1)$ are exactly those from $\boldsymbol{\alpha}_1$ that correspond to the ground truth class of the given test sample.

From Figure 12.2, we see that σ_{acc} was generally much larger than σ_{mc}, and that the Kernel SRC method could tolerate substantial ℓ^1 and ℓ^2-support error before classification deteriorated. Further, perfect classification was achieved even for maximally dense $\boldsymbol{\alpha}_1$. This suggests that a strictly-sparse solution vector is not necessary to the success of SRC.

As the level of noise η increased, we see in Figure 12.2 that σ_{acc} decreased towards σ_{mc}. However,

$$\lim_{\eta \to \infty} \sigma_{\text{acc}} \neq \sigma_{\text{mc}}.$$

Once the class structure was lost due to noise in the original space, increasing the noise level further had no effect on the quantities displayed in Figure 12.2. In other words, Figure 12.2c is representative of the results for larger values of η.

We also observe that for $\eta = 0.001$, the sparsest solution was obtained by ℓ^1-minimization for values of σ slightly larger than σ_{mc} (note the position of the σ_{mc} arrow tip in Figure 12.2a). In fact, the mutual coherence of the data set with $\eta = 0.001$ reached $\mu = 0.9994$ before ℓ^1-minimization failed to retrieve the sparsest solution. This indicates that when the classes are well-separated (for small η and sufficiently small σ, separability in the original space carries over to kernel space in

this experiment), ℓ^1/ℓ^0-equivalence can still be achieved even when the mutual coherence is much larger than that allowed by Eq. (9.4). This reinforces the findings from the last chapter, namely, that ℓ^1/ℓ^0-equivalence holds on highly-correlated data as long as the vectors corresponding to the support of the sparsest solution are sufficiently separated from the other dictionary elements. On the other hand, for larger values of η, i.e., when the classes were less well-separated, the bound in Eq. (9.4) appears to be approximately tight.

12.4.5. Examining the Accuracy Threshold. It is notable that σ_{acc} is substantially larger than σ_{mc} for all η, and that the accuracy in Kernel SRC has a steep drop-off as soon as $\sigma > \sigma_{\text{acc}}$. The value σ_{acc} appears to be a threshold for which the linear relationship between $\phi(\boldsymbol{y})$ and the training samples in its ground truth class cannot be identified by the classification mechanism in (Kernel) SRC. We want to know what triggers this threshold.

We first look for an "elbow" or sharp change in the correlation between $\phi(\boldsymbol{y})$ and training samples in its ground truth class, and that between $\phi(\boldsymbol{y})$ and samples in other classes. In particular, we computed

$$\text{corr}_{\text{GT}} := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{\boldsymbol{x}_j^{(l)}:\, \boldsymbol{y} \in \text{ class } l}{\text{median}} \left\langle \phi(\boldsymbol{y}), \phi(\boldsymbol{x}_j^{(l)}) \right\rangle \right\}$$

and

$$\text{corr}_{\text{other}} := \underset{\text{all trials}}{\text{mean}} \left\{ \underset{l:\, \boldsymbol{y} \notin \text{ class } l}{\text{median}} \left\{ \underset{1 \le j \le N_l}{\text{median}} \left\langle \phi(\boldsymbol{y}), \phi(\boldsymbol{x}_j^{(l)}) \right\rangle \right\} \right\}.$$

Again, we compute the median quantities within each trial to make the correlation values more robust to sample outliers.

The results for $\eta = 0.1$ are shown in Figure 12.3. The plots for the other values of η are similar. As we can see, the accuracy threshold σ_{acc} occurred *after* the sharp increase in the correlation quantities. In fact, we see that SRC was able to retrieve the correct classification assignment when corr_{GT} was only moderately larger than $\text{corr}_{\text{other}}$. On the other hand, the sharp increase in the correlation quantities appears to correspond to the steep increase in sparsity level, which makes sense in the context of the mutual coherence equivalence guarantee in Theorem 9.2.4.

As a more informative approach to understanding the accuracy threshold, in particular, what causes the sharp drop-off in accuracy at σ_{acc}, we consider the distribution of the absolute values of the coefficients, i.e., the magnitude of the coordinates of $\boldsymbol{\alpha}_1$, with respect to the different classes. In particular, for $\eta = 0.1$, we computed the mean vector $|\boldsymbol{\alpha}_1|$ over the $N_0 = 5$ class $l = 20$ training samples, and then averaged the result over 100 trials:

$$\underset{\text{all trials}}{\text{mean}} \left\{ \underset{\boldsymbol{y} \in \text{ class } l=20}{\text{mean}} \left\{ |\boldsymbol{\alpha}_1| \right\} \right\}.$$

We lastly normalized the resulting vector so that its entries summed to 1.

We plot the results in Figure 12.4 for a handful of representative values of σ. The x-axis in the left-hand-side plots (Figures 12.4a, 12.4c, 12.4e, and 12.4g) corresponds to the individual coordinates of the averaged vector $|\boldsymbol{\alpha}_1| \in \mathbb{R}^{N_{\text{tr}}}$. The coordinates corresponding to training samples in each class are simply summed to produce the right-hand-side plots (Figures 12.4b, 12.4d, 12.4f, and 12.4h), so that the contribution from each class in the representation of $\phi(\boldsymbol{y})$ can be viewed easily. We also include the corresponding Kernel SRC classification accuracies for reference.

Given the dominance of coefficients corresponding to class $l = 20$ in Figures 12.4a-12.4d, it is not surprising that Kernel SRC obtains perfect accuracy in these cases. It is also quite clear from these figures that small coefficients in the wrong class do not negatively affect classification accuracy. Thus there is no reason to require a solution sparser than that with $\sigma = 3$.

For $\sigma \in \{5, 9\}$, the closeness in the coefficient magnitudes between those corresponding to class $l = 20$ and those corresponding to other classes illustrates the decreased accuracy in Kernel SRC; recall that these plots contain averages. Additionally, we note that the distribution of the coefficients in class $l = 20$ became fairly unbalanced among that class's training samples for these large values of σ. This is because as mutual coherence increased, the class $l = 20$ samples became more and more parallel to each other. Thus most of $\phi(\boldsymbol{y})$ could be represented using only the first training sample in that class.

Figure 12.4 helps to explain the sharp drop-off in accuracy at σ_{acc}. Though the quantities corr_{GT} and $\text{corr}_{\text{other}}$ are only slightly increasing at σ_{acc} (and the general behavior of the coefficients varying smoothly), the threshold occurs right at the point that the coefficients of other classes become competitive with those from the correct class (as we would expect). The sharp drop-off can be

attributed to the nonlinearity of the `min` function in determining $\min_{1\leq l\leq L}\{\|\phi(\boldsymbol{y})-\Phi(X_{\text{tr}})\delta_l(\boldsymbol{\alpha}_1)\|_2\}$ in the classification stage of (Kernel) SRC.

12.5. Key Findings

First of all, we acknowledge that this experiment did not satisfy all of the goals laid out in its introduction. However, we can still draw some important conclusions:

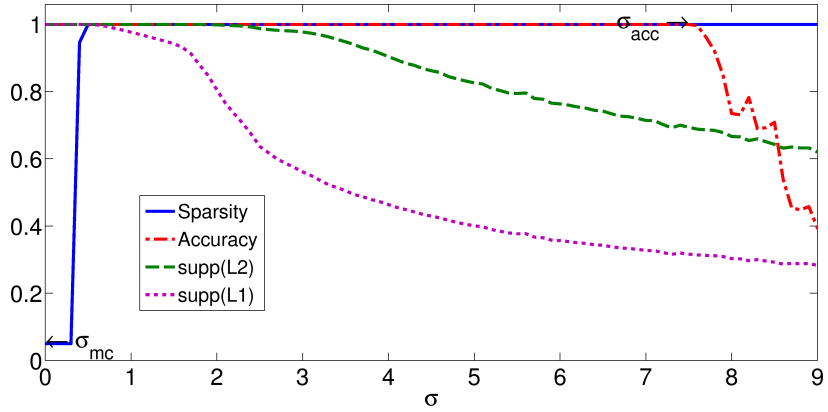
- Any procedure that spreads out the data in each class in a way that decreases mutual coherence yet aims to maintain class structure will necessarily come into conflict with maintaining a linear relationship between $\boldsymbol{y}$ and *any* subset of training samples. More precisely, it is generally impossible to write $\boldsymbol{y}$ as a linear combination of the training samples in class l while satisfying the bound

$$\|\boldsymbol{\alpha}\|_0 < \frac{1}{2}\Big(1+\frac{1}{\mu(X_{\text{tr}})}\Big) \approx \frac{1}{2}\Big(1+\frac{1}{\mu([X^{(l)},\boldsymbol{y}])}\Big),$$

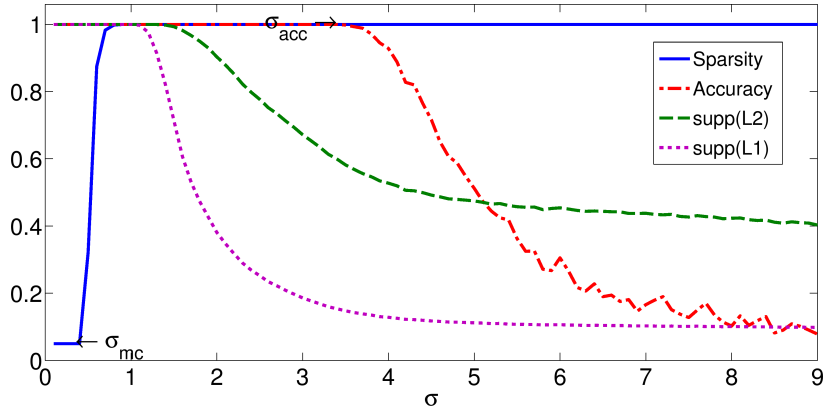
  i.e., when $\boldsymbol{y}$ is spread out in the same manner as the other samples in the database. Besides artificially generating $\boldsymbol{y}$ as a linear combination of the training samples *after* they have been spread out, it is not clear to us how to overcome this conflict.
- Though generating $\boldsymbol{y}$ as a linear combination of its ground truth class training samples in kernel space prevented us, in some sense, from isolating the relationship between σ_{mc} and classification accuracy, we were still able to study the correspondence between σ_{mc} and sparsity level $\|\boldsymbol{\alpha}_1\|_0$. In particular, we confirmed our previous findings that perfect recovery can be achieved on highly-correlated data as long as the classes are sufficiently well-separated (in this experiment, this meant small η).
- We saw that there was a sharp drop-off in classification accuracy at $\sigma_{\text{acc}} > \sigma_{\text{mc}}$, which was not directly correlated with a sharp change in either sparsity or the relationship between within-class and between-class correlation, or in the normalized ℓ^2 and ℓ^1-norms of $\delta_{\text{GT}}(\boldsymbol{\alpha}_1)$. Though ℓ^1/ℓ^0-equivalence (whether provable by Theorem 9.2.4 or not) was a way to ensure perfect classification accuracy in this experiment, it was not necessary. The classification mechanism in SRC can clearly tolerate even the maximal number of nonzero coefficients in the representation, as long as the magnitudes of coefficients corresponding

to the wrong classes are small with respect to those from the correct class. In this sense, *relative*—or *approximate*—sparsity is the key to SRC. It might be possible to make this idea precise in terms of a coefficient thresholding procedure similar to the one used in Chapter 11.

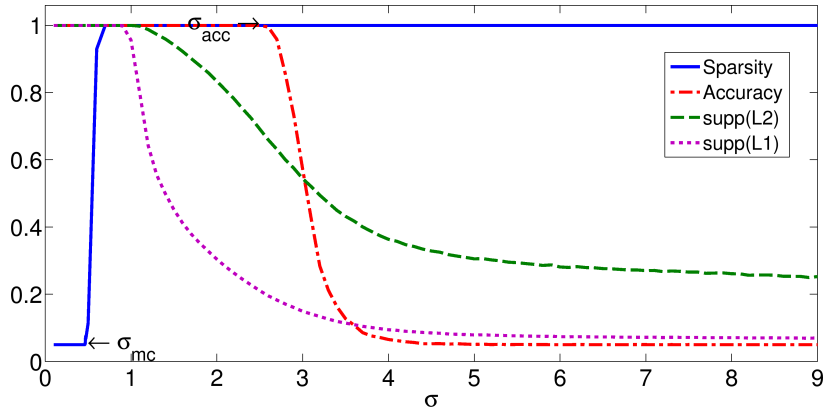
In future research, it would be interesting to consider the modification of the above experiment when noise is added to the test sample $\phi(\boldsymbol{y})$ after it is generated as a linear combination of its ground truth class training samples in kernel space. Of course, this will not have the same effect as adding noise to the original (and implicitly-defined) test sample $\boldsymbol{y}$, but it would allow us to investigate the relationship between classification accuracy in SRC and the mutual coherence bound in the case of noise as stated in Theorem 9.2.5.



(a) $\eta = 0.001$



(b) $\eta = 0.1$



(c) $\eta = 0.5$

FIGURE 12.2. Average sparsity, accuracy, supp(ℓ^2) and supp(ℓ^1) (over 100 trials) as σ increased in the kernel setup. The annotations "σ_{mc}" and "σ_{acc}" denote the maximum σ for which Eq. (9.4) holds and for which maximum accuracy is obtained in Kernel SRC, respectively.

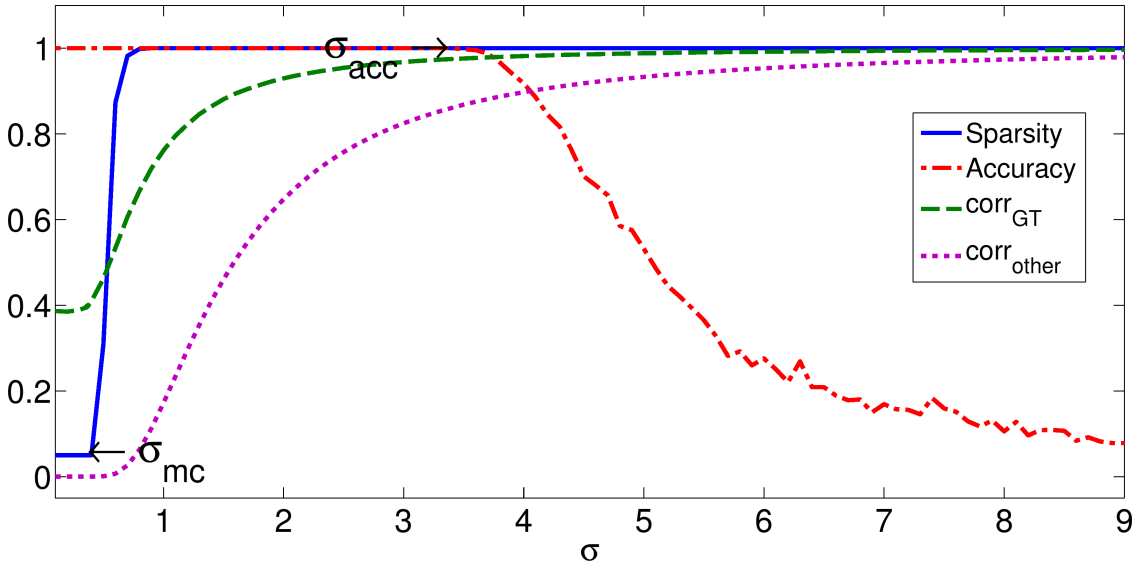
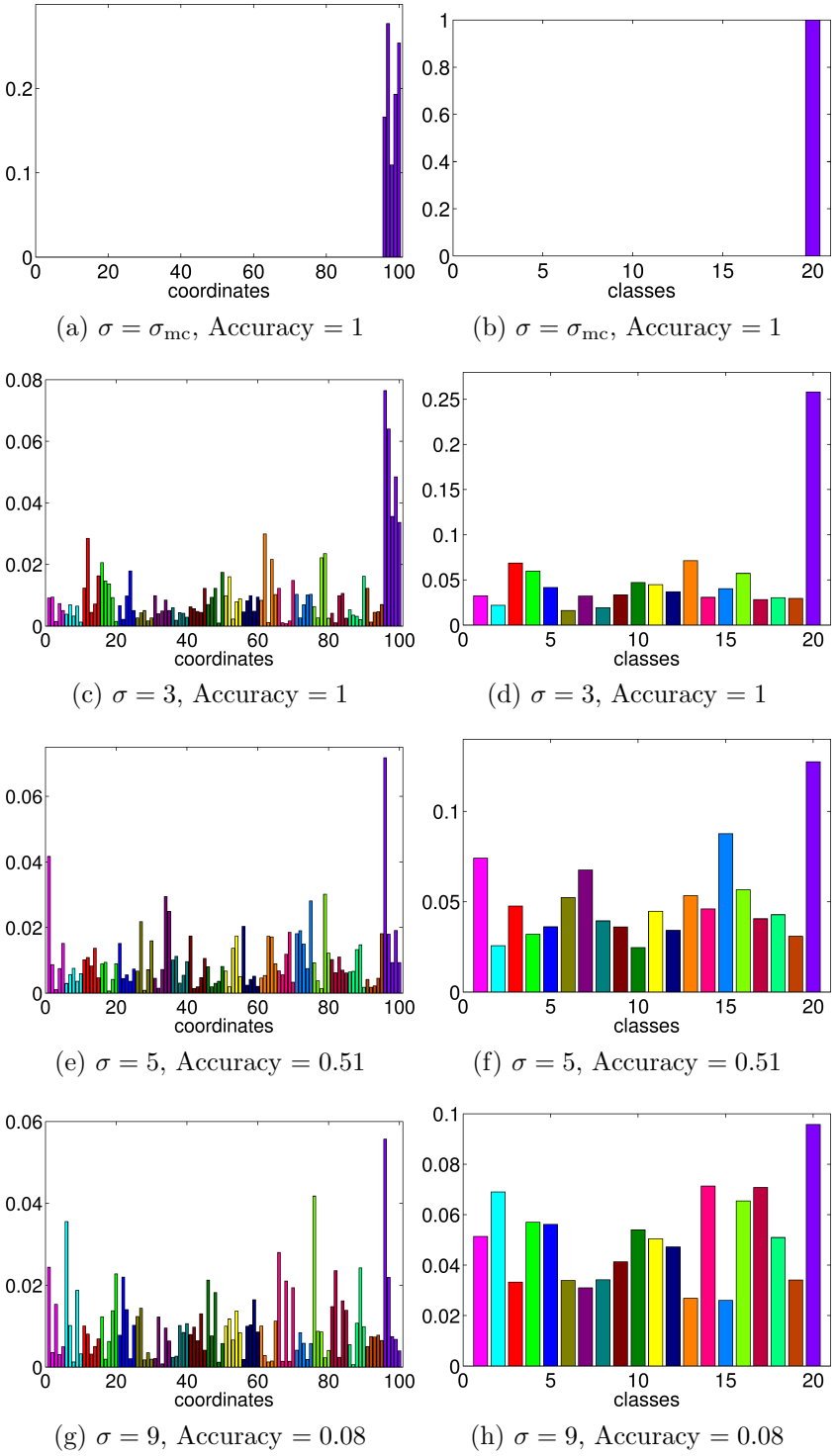


FIGURE 12.3. Median correlation (averaged over 100 trials) between the test sample $\phi(\boldsymbol{y})$ and training samples in the same class (corr_{GT}) and training samples in different classes ($\text{corr}_{\text{other}}$) for the synthetic database with $\eta = 0.1$. Sparsity and accuracy are also displayed for comparison. Notice that the drop in accuracy occurs well after the jump in the correlation terms and sparsity.



(a) $\sigma = \sigma_{\text{mc}}$, Accuracy $= 1$ (b) $\sigma = \sigma_{\text{mc}}$, Accuracy $= 1$

(c) $\sigma = 3$, Accuracy $= 1$ (d) $\sigma = 3$, Accuracy $= 1$

(e) $\sigma = 5$, Accuracy $= 0.51$ (f) $\sigma = 5$, Accuracy $= 0.51$

(g) $\sigma = 9$, Accuracy $= 0.08$ (h) $\sigma = 9$, Accuracy $= 0.08$

FIGURE 12.4. Average class contributions (over 100 trials) of coefficient vectors corresponding to class $l = 20$ test samples. The colors denote the classes.

CHAPTER 13

Conclusion of Part 2

In Part 2 of this dissertation, we investigated the applicability of ℓ^1/ℓ^0-equivalence guarantees on dictionaries containing training samples. We detailed the inherent conflict between tightly-clustered classes—desirable for good classification—and the sufficient incoherence required by equivalence guarantees such as those based on restricted isometry and mutual coherence. In particular, we proved that under the assumptions of SRC, i.e., that class manifolds are linear subspaces spanned by their respective training data, Donoho et al.'s mutual coherence guarantees can only hold in the case that we have *exactly* enough training samples to span each lower-dimensional subspace. Considering that the performance of SRC generally improves as the training class size increases (recall that a primary goal of the LPCA-SRC algorithm from Part 1 was to enlarge the size of the training set in SRC), it is likely counter-productive for classification purposes to restrict the training set in this way. Further, despite the methods referenced in Part 1 to estimate the class manifold dimension, it is impractical to assume that such approaches will always work perfectly.

Despite not being able to prove ℓ^1/ℓ^0-equivalence on most class-structured data, we saw that it can indeed be achieved in some specific cases. Inspired by the random model of Wright and Ma to generate face image-like databases, we designed an experiment to test the ability of ℓ^1-minimization to recover the sparsest solution on highly-correlated data. The results were mostly positive. We observed that in all cases, ℓ^1-minimization recovered a solution closely approximating the sparsest solution (defined by generating the test sample as a linear combination of training samples in its ground truth class). Further, within-class correlation actually improved recovery relative to uniformly-random data, provided that the between-class correlation was sufficiently low, i.e., that the classes were sufficiently separated. In many cases, ℓ^1-minimization exactly recovered the sparsest solution. Additionally, in the case that noise was added to the test sample, the correct support was found in nearly every case in which correlation was introduced.

We also considered the role of sparsity in the context of SRC and similar classification algorithms. One obstacle in determining this relationship is obtaining access to the sparsest solution for comparison without the aid of ℓ^1/ℓ^0-equivalence guarantees. Towards resolving this problem, we designed a nonlinear transform, based on kernel methods using the Gaussian kernel, to decrease the within-class mutual coherence while still maintaining class structure so that (hypothetically) provable equivalence and good classification could be simultaneously achieved. However, we found that the degree to which we had to decrease coherence in this setup meant that the test sample was no longer in the span of the training data, and so we were forced to limit our analysis to test samples artificially generated as linear combinations of their ground truth class training samples, as in Chapter 11. Though this to some extent limited the applicability of our experiment, the results clearly indicate that strict sparsity is not necessary for good classification in SRC. Instead, its success lies in its ability to correctly differentiate the coefficient magnitudes of training samples in different classes, i.e., to find *approximately* or *relatively* sparse solutions, in the case that the linear subspace assumption is observed and the classes themselves are not too correlated, i.e., not close together.

There is certainly much work to be done to quantify these findings. We mention two potential next steps: Eldar and Kuppinger's notion of *block-coherence* [**36**], with blocks corresponding to classes of the training database, might serve to make precise the meaning of between-class correlation; note that this was observed to play a role in both ℓ^1/ℓ^0-equivalence on highly-correlated data and SRC's classification performance. Additionally, the accuracy threshold detected in Chapter 12 might be better understood in the context of Wang et al.'s interpretation of SRC as a maximum margin-based classifier [**98**]. As an alternative to the thresholding route as suggested in Chapter 12, their work could be very helpful in rigorously defining the concept of *approximate sparsity* as it relates to the classification performance of SRC.

CHAPTER 14

Final Remarks

In this last chapter, we make some final comments regarding the relationship between classification and sparsity.

We saw on the sinusoidal waveform synthetic database in Chapter 5 that the sparsity enforced via ℓ^1-minimization resulted in an extremely local approximation of the test sample, leading to good classification performance that improved as the sampling density of the class manifolds increased. In this case, *sparsity implied locality*, and as we witnessed via the ℓ^2-regularized LDSL-SRC results, the converse was not true, at least to the extent necessary for good classification performance on this database.

However, the ℓ^1-minimized solution does not always result in a local approximation of the test sample: On the ORL database, for instance, we saw that the coefficient vector contained significant coefficients from nonlocal training samples, as evidenced by the increased accuracy of $\text{SRC}_{\text{pruned}}$ over SRC. This demonstrates that enforcing locality via a simple dictionary pruning step can improve classification performance over ℓ^1-minimization alone.

These examples illustrate that neither sparsity nor locality alone is enough to produce good classification results in all situations. The algorithm LPCA-SRC is a step in combining both properties into a single classifier; however, there is certainly room for improvement in terms of computational complexity and robustness to data noise because of the use of tangent vectors. There is also more work to be done in terms of the easy estimation of tangent vector error and providing the user with concrete recommendations and ideally, performance guarantees.

Given the findings in Part 2, we are confident that ℓ^1-minimization, even on highly-correlated data, will find a sparse or relatively-sparse coefficient vector provided that the test sample can be approximated reasonably well using samples from a single class and that the training set contains enough class structure to make the classification problem viable. This is despite the fact that the mutual coherence ℓ^1/ℓ^0-equivalence guarantees—the only tractable approach discussed in Chapter

9—provably cannot be applied in general, provided that the assumptions in SRC are met. We observed that, though the exact sparsest solution may be found using ℓ^1-minimization provided that the classes are well-separated, this may be unnecessary, due to the robust classification mechanism in SRC. Further, the relatively sparse solutions awarded by ℓ^1-minimization can lead to significantly better classification results than the dense solutions obtained using ℓ^2-regularization, particularly when the feature dimension is small or, as our experiments in Chapter 7 suggest, in the case that multiple good per-class approximations of the test sample exist, as on the sinusoidal waveform synthetic database. More work is needed to gain deeper understanding of these relationships.

Our results are very much dependent on the representation-based approach to classification being used, and it is important to note that this method is not applicable on some data sets. For example, modeling a test sample as an approximate linear combination of its same-class training samples is not appropriate when the training data do not lie on smooth manifolds (or not on manifolds at all) or when there is only a single training sample per class.[1] Further, we have focused very little on the fact that these methods require the training samples to be normalized; when essential class-structure is contained in the norms of the samples, these methods fail.[2]

We end this dissertation by mentioning the *bet on sparsity* principle of Hastie et al. [**48**]. In the context of variable selection for regression and classification problems, this principle states that we should use a procedure that will find sparse solutions (i.e., we should assume a sparseness prior), because *no procedure will do well if the solution is dense* [**48**]. That is, consider the dictionary with columns representing variables (or features) and rows representing samples. (Recall that the opposite is true in the representation-based classification scenario.) Given that there are more variables than samples so that the resulting system is underdetermined, the goal is to express a vector of responses $\boldsymbol{y}$ as an approximate linear combination of these variables. If the (ground truth) coefficients of this approximation are not sparse, then we do not have enough samples to correctly identify them [**48**]. For example, both ℓ^1-regularization and the denser ℓ^2-regularization approaches will fail to recover this solution. So we might as well assume that the coefficients are sparse and move forward.

[1] However, SRC has been extended to the single sample face recognition scenario. See Deng et al.'s algorithm [**26**].
[2] Interestingly, the kernel trick has been used to address this problem. See Zhang et al.'s paper [**116**].

Let us detail an analogous principle in the representation-based classification scenario. Given that we wish to express a test sample $\boldsymbol{y}$ over a training set $X_{\text{tr}} \in \mathbb{R}^{m \times N_{\text{tr}}}$ with $m \ll N_{\text{tr}}$, then if $\boldsymbol{y}$ is not an approximate linear combination of its same-class training samples, it is unclear how we will be able to determine the correct class using any representation-based classifier. On the other hand, if a representation using the correct class exists and is not sparse, i.e., if L is small, then either the sparsest representation will consist of a subset of these class coefficients (in which case this class will be identified), or the sparsest representation will contain nonzero coefficients corresponding to training samples from several cases. If this happens, then there is no reason to think (in general) that an incorrect class will contribute more to this representation than the correct one (barring outliers and heavy sample corruption, of course). Even in the case that $\boldsymbol{y}$ lies in the intersection of two class manifolds, we will at worst be aware of the uncertainty of the classification decision via the similarity in class residuals $\text{err}_l(\boldsymbol{y})$ and have identified the two most likely classes. Note that, in this scenario, the block-sparse SSR methods would do no better, and CRC-RLS may do substantially worse.

Thus, in considering representation-based classification, one should "bet on sparsity" via ℓ^1-minimization. Even in the case that the class approximation is not found, it is unlikely that any other approach to finding it will do better.

APPENDIX A

Proof of Tangent Bound Modification

The main result in Kaslovsky and Meyer's paper [**57**] bounds the error between an approximated tangent plane computed using standard local PCA and the true tangent plane. We refer to this as the "tangent bound." One can view the offline portion of the classification algorithm LPCA-SRC as a method of generating new training samples in the form of shifted and scaled basis vectors of the approximated tangent plane; we call these basis elements "tangent vectors." Notably, LPCA-SRC does not use standard local PCA to compute the tangent vectors but instead the local PCA technique of Singer and Wu [**87**]. Thus, in order to use the tangent bound to estimate the error in the newly-generated training samples (i.e., the tangent vectors), as was done in Chapter 6 above, the bound in the Kaslovsky and Meyer paper [**57**] must be amended to account for the difference in local PCA implementations. We do this here.

A.1. Tangent Bound Details

A.1.1. Geometric Setup. Given a smooth, d-dimensional manifold $\mathcal{M} \subseteq \mathbb{R}^m$ and points randomly sampled from $\mathcal{M}$, we can approximate the tangent (hyper)plane to the manifold at a sampled point $\boldsymbol{x}_0$. Local PCA is a standard tool to do this. To perform standard local PCA, we determine the n nearest neighbors of $\boldsymbol{x}_0$ from the sampled data set and compute the covariance matrix from these samples. The first d eigenvectors of the covariance matrix are then used as the basis vectors of the approximated tangent plane.

Kaslovsky and Meyer's tangent bound [**57**] determines (with high probability) the maximum angle between the true tangent plane and the approximated tangent plane. An important part of the authors' theory is a specific *geometric setup*, involving a (theoretical) shift and rotation of the manifold under which this angle is invariant. The geometric setup identifies $\boldsymbol{x}_0$ with the origin and aligns the principal axes of curvature of $\mathcal{M}$ at $\boldsymbol{x}_0$ with the coordinate axes. This allows for the separation of the linear and curvature components of each of the n neighbors of $\boldsymbol{x}_0$. That is, a

nearby point $\boldsymbol{x}$ in the shifted and rotated coordinate system can be written as

$$\boldsymbol{x} = \boldsymbol{\ell} + \boldsymbol{c} + \boldsymbol{e} = \begin{bmatrix} \ell_1 \\ \vdots \\ \ell_d \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ c_{d+1} \\ \vdots \\ c_m \end{bmatrix} + \begin{bmatrix} e_1 \\ \vdots \\ e_d \\ e_{d+1} \\ \vdots \\ e_m \end{bmatrix} \in \mathbb{R}^m,$$

where the vectors $\boldsymbol{\ell}$, $\boldsymbol{c}$, and $\boldsymbol{e}$ refer to the linear, curvature, and noise components of $\boldsymbol{x}$, respectively.

If the n nearest neighbors of $\boldsymbol{x}_0$ in the data set obey certain sampling assumptions, i.e., if their linear components are assumed to be distributed uniformly within a neighborhood of $\boldsymbol{x}_0$ on the tangent plane at $\boldsymbol{x}_0$, then we can model this situation as follows: For $\boldsymbol{x} = \boldsymbol{\ell}+\boldsymbol{c}+\boldsymbol{e}$ in the geometric setup, the first d coordinates of $\boldsymbol{\ell} \in \mathbb{R}^m$ can be viewed as random samples from the d-dimensional tangent plane at $\boldsymbol{x}_0$, i.e., the span of the first d axes, uniform within $B^d_{\mathbf{0}}(r)$, the d-dimensional ball of radius r centered at $\boldsymbol{x}_0 = \mathbf{0}$. Here, r is used to define the local neighborhood, and we stress that r bounds the distance between $\boldsymbol{x}_0$ and $\boldsymbol{x}$ as measured *along the tangent plane*, i.e., $\|\boldsymbol{\ell}\| \leq r$.

The curvature component $\boldsymbol{c} \in \mathbb{R}^m$ can be viewed as having nonzero coordinates defined by

$$c_i := \frac{1}{2}\Big(\kappa_1^{(i)}\ell_1^2 + \cdots + \kappa_d^{(i)}\ell_d^2\Big), \quad i = d+1, \ldots, m, \tag{A.1}$$

where $\{\kappa_1^{(i)}, \ldots, \kappa_d^{(i)}\}$ are the principal curvatures of $\mathcal{M}$ at $\boldsymbol{x}_0$ in the ith normal direction, $i = d+1, \ldots, m$.[1] Lastly, the noise component $\boldsymbol{e} \in \mathbb{R}^m$ can be viewed as being drawn from $\mathcal{N}(\mathbf{0}, \sigma^2 I_m)$ (assuming that this is an appropriate characterization of the sample noise).

A.1.2. Notation. We assume throughout this discussion that local data samples can be decomposed as above, i.e., we view the samples as points in the shifted and rotated coordinate system, generated as above. Given the local samples $\boldsymbol{x}^{(k)} = \boldsymbol{\ell}^{(k)} + \boldsymbol{c}^{(k)} + \boldsymbol{e}^{(k)} \in \mathbb{R}^m$, $1 \leq k \leq n$, define the

[1]The reader may recall from the tangent distance theorem in Chapter 6 that Eq. (A.1) is actually the second-order Taylor expansion of the ith coordinate of the curvature component. Citing the numerical results of Tyagi et al. [**94**], Kaslovsky and Meyer claim that this expansion is sufficiently general in this context [**57**].

matrix $X := [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}] \in \mathbb{R}^{m \times n}$. Then we can write

$$X = L + C + E,$$

where $L := [\boldsymbol{\ell}^{(1)}, \ldots, \boldsymbol{\ell}^{(n)}]$ is the linear/tangent component, $C := [\boldsymbol{c}^{(1)}, \ldots, \boldsymbol{c}^{(n)}]$ is the curvature component, and $E := [\boldsymbol{e}^{(1)}, \ldots, \boldsymbol{e}^{(n)}]$ is the noise component. To approximate the tangent plane, we aim to recover L from X. As mentioned, standard local PCA's solution to this problem is to compute the first d eigenvectors of the covariance matrix of X and set these to be the approximated tangent plane basis vectors. Kaslovsky and Meyer's tangent bound compares this solution to the true tangent plane.

Let $\widetilde{X} := X - (1/n)\sum_{k=1}^{n} \boldsymbol{x}^{(k)}$ be the *centered* version of X. Kaslovsky and Meyer define the following eigendecompositions [**57**]:

$$\frac{1}{n}\widetilde{X}\widetilde{X}^{\mathsf{T}} = \widehat{U}\,\widehat{\Lambda}\,\widehat{U}^{\mathsf{T}} = \begin{bmatrix}\widehat{U}_1 & \widehat{U}_2\end{bmatrix} \begin{bmatrix}\widehat{\Lambda}_1 & 0 \\ 0 & \widehat{\Lambda}_2\end{bmatrix} \begin{bmatrix}\widehat{U}_1 & \widehat{U}_2\end{bmatrix}^{\mathsf{T}}$$

and

$$\frac{1}{n}\widetilde{L}\widetilde{L}^{\mathsf{T}} = U\,\Lambda\,U^{\mathsf{T}} = \begin{bmatrix}U_1 & U_2\end{bmatrix} \begin{bmatrix}\Lambda_1 & 0 \\ 0 & \Lambda_2\end{bmatrix} \begin{bmatrix}U_1 & U_2\end{bmatrix}^{\mathsf{T}},$$

where $\widehat{U}_1$ and U_1 contain the first d columns of $\widehat{U}$ and U, respectively, and the diagonal matrices of eigenvalues, $\widehat{\Lambda}$ and Λ, are similarly divided. It is assumed that the eigenvalues are in decreasing order. It follows that the angle between the true and approximated tangent planes can be measured by computing the Frobenius norm of the difference between their respective orthogonal projectors:

$$\left\|\widehat{U}_1\widehat{U}_1^{\mathsf{T}} - U_1U_1^{\mathsf{T}}\right\|_F. \tag{A.2}$$

Note that $\widehat{U}_1$ contains the approximated tangent plane basis vectors found by standard local PCA.

To bound $\left\|\widehat{U}_1\widehat{U}_1^{\mathsf{T}} - U_1U_1^{\mathsf{T}}\right\|_F$, Kaslovsky and Meyer [**57**] expand the covariance matrix of X as

$$\frac{1}{n}\widetilde{X}\widetilde{X}^{\mathsf{T}} = (\widetilde{L} + \widetilde{C} + \widetilde{E})(\widetilde{L} + \widetilde{C} + \widetilde{E})^{\mathsf{T}} = \widetilde{L}\widetilde{L}^{\mathsf{T}} + \Delta,$$

where

$$\Delta := \frac{1}{n}(\widetilde{C}\widetilde{C}^{\mathsf{T}} + \widetilde{E}\widetilde{E}^{\mathsf{T}} + \widetilde{L}\widetilde{C}^{\mathsf{T}} + \widetilde{C}\widetilde{L}^{\mathsf{T}} + \widetilde{L}\widetilde{E}^{\mathsf{T}} + \widetilde{E}\widetilde{L}^{\mathsf{T}} + \widetilde{C}\widetilde{E}^{\mathsf{T}} + \widetilde{E}\widetilde{C}^{\mathsf{T}}), \tag{A.3}$$

and $\widetilde{L}$, $\widetilde{C}$, and $\widetilde{E}$ denote the centered versions of L, C, and E, respectively. The authors next compute a lower bound for λ_d, the dth largest eigenvalue of $(1/n)\widetilde{L}\widetilde{L}^{\mathsf{T}}$ (equivalently, the (d,d)th entry of Λ_1), and upper bounds for the matrix norms $\|U_1^{\mathsf{T}}\Delta U_1\|_F$, $\|U_2^{\mathsf{T}}\Delta U_2\|_F$, $\|U_2^{\mathsf{T}}\Delta U_1\|_F$, and $\|U_1^{\mathsf{T}}\Delta U_2\|_F$. They then use these quantities in the following theorem:

THEOREM A.1.1 (Davis and Kahan [**25**], Stewart [**89**]). *Set*

$$\delta := \lambda_d - \left\|U_1^{\mathsf{T}}\Delta U_1\right\|_F - \left\|U_2^{\mathsf{T}}\Delta U_2\right\|_F.$$

Define the following conditions:

- *Condition 1:* $\delta > 0$,
- *Condition 2:* $\left\|U_1^{\mathsf{T}}\Delta U_2\right\|_F \left\|U_2^{\mathsf{T}}\Delta U_1\right\|_F < \frac{1}{4}\delta^2$.

If both conditions hold, then

$$\left\|\widehat{U}_1\widehat{U}_1^{\mathsf{T}} - U_1U_1^{\mathsf{T}}\right\|_F \leq 2\sqrt{2}\,\frac{\left\|U_2^{\mathsf{T}}\Delta U_1\right\|_F}{\delta}.$$

A.1.3. Sampling Assumptions. Kaslovsky and Meyer [**57**] assume that the number of neighbors n satisfies $n \geq O(d\log d)$. More generally, the spirit of their analysis is based on a generous sampling density, something to keep in mind when considering the applicability of (our modification to) their theorem in the context of classification problems.

A.2. Singer and Wu's Local PCA

In standard local PCA, the approximated tangent plane basis vectors are found by computing the eigenvectors of the covariance matrix $(1/n)\widetilde{X}\widetilde{X}^{\mathsf{T}}$. However, the local PCA technique of Singer and Wu [**87**] computes the eigenvectors of a slightly different covariance matrix. Given the local sample matrix $X = [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}]$, we define

$$Y := (X - [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_0])D_0,$$

for $\boldsymbol{x}_0$ the point on the manifold at which we are computing the tangent plane. D_0 is a diagonal matrix whose (j,j)th entry contains an increasing function of the inverse distance between $\boldsymbol{x}_0$ and $\boldsymbol{x}^{(j)}$, $1 \leq j \leq n$, so that points nearer to $\boldsymbol{x}_0$ contribute more to the construction of the tangent plane at $\boldsymbol{x}_0$. The approximated tangent plane basis vectors in Singer and Wu's local PCA are given by the eigenvectors of the matrix YY^{T}, equivalently, the left singular vectors of Y. Consider the difference between this method and standard local PCA, which computes the eigenvectors of $(1/n)\widetilde{X}\widetilde{X}^{\mathsf{T}}$, where $\widetilde{X} = (X - [\boldsymbol{\mu}, \ldots, \boldsymbol{\mu}])I$ and $\boldsymbol{\mu}$ is the mean column of X given by $\boldsymbol{\mu} := (1/n)\sum_{k=1}^{n} \boldsymbol{x}^{(k)}$.

A.3. Modifying Kaslovsky and Meyer's Setup

A.3.1. Amending the Geometric Setup. What effect does the shift by $\boldsymbol{x}_0$ in Singer and Wu's local PCA, in contrast to the shift by $\boldsymbol{\mu}$ in standard local PCA, have on the geometric setup and the model it produces? Since the geometric setup automatically (i.e., *inherently*) shifts the points by $\boldsymbol{x}_0$ in order to treat this point as the origin, it suffices to consider the *uncentered* matrix X in the tangent bound analysis. Thus the Singer and Wu case with $D_0 = I$ is simply Kaslovsky and Meyer's original setup with X replacing the centered matrix $\widetilde{X}$. In other words, we only need to consider $Y = XD_0$ in the geometric setup and do not need to make any explicit compensation for the centering by $\boldsymbol{x}_0$.

Next we consider the effect of the weight matrix D_0. Let D_0 have diagonal entries $d_1, \ldots, d_n$. We expand

$$Y = (X - [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_0])D_0 = [d_1(\boldsymbol{x}^{(1)} - \boldsymbol{x}_0), \ldots, d_n(\boldsymbol{x}^{(n)} - \boldsymbol{x}_0)].$$

Thus when $\boldsymbol{x}_0$ is treated as the local origin, the points we must consider are the scaled samples

$$\{d_1\boldsymbol{x}^{(1)}, \ldots, d_n\boldsymbol{x}^{(n)}\}.$$

Recall that in local PCA in the LPCA-SRC algorithm (i.e., Algorithm 4), the samples in X are sorted in increasing order of distance to $\boldsymbol{x}_0$. However, this is purely for convenience: we will obtain the same eigenvectors regardless of the arrangement of the columns of X. Thus d_k should refer to the weight of the kth sample $\boldsymbol{x}^{(k)}$, which may not necessarily be the kth closest sample to $\boldsymbol{x}_0$.

Next we consider how to model (or view as randomly-generated samples) the points $d_1\boldsymbol{x}^{(1)}, \ldots, d_n\boldsymbol{x}^{(n)}$ in the geometric setup. We have

$$d_k\boldsymbol{x}^{(k)} = d_k(\boldsymbol{\ell}^{(k)} + \boldsymbol{c}^{(k)} + \boldsymbol{e}^{(k)}) = d_k\boldsymbol{\ell}^{(k)} + d_k\boldsymbol{c}^{(k)} + d_k\boldsymbol{e}^{(k)},$$

for $1 \leq k \leq n$. That is, the linear, curvature, and noise components are simply scaled by d_k. However, the (weighted) linear components $d_k\boldsymbol{\ell}^{(k)}$ are no longer uniformly distributed within $B_{\mathbf{0}}^d(\boldsymbol{x}_0)$. Of course, this distribution depends on d_k.

In the LPCA-SRC algorithm, we set

$$\begin{aligned} d_k :=& \sqrt{1 - \frac{\|\boldsymbol{x}^{(k)}\|^2}{s}} \\ =& \sqrt{1 - \frac{\|\boldsymbol{\ell}^{(k)}\|^2 + \|\boldsymbol{c}^{(k)}\|^2 + \|\boldsymbol{e}^{(k)}\|^2 + 2\left\langle \boldsymbol{\ell}^{(k)} + \boldsymbol{c}^{(k)}, \boldsymbol{e}^{(k)} \right\rangle}{s}}, \end{aligned}$$

where s is a scaling factor satisfying $s > \max\limits_{1\leq k\leq n} \|\boldsymbol{x}^{(k)}\|^2$. The second equality follows from

$$\left\langle \boldsymbol{\ell}^{(k)}, \boldsymbol{c}^{(k)} \right\rangle = 0.$$

A.3.2. Amending the Notation. We write $XD_0 = LD_0 + CD_0 + ED_0$ and

$$\frac{1}{n}XD_0(XD_0)^{\mathsf{T}} = \frac{1}{n}(LD_0 + CD_0 + ED_0)(LD_0 + CD_0 + ED_0)^{\mathsf{T}} = \frac{1}{n}LD_0(LD_0)^{\mathsf{T}} + \Theta,$$

where

$$\begin{aligned} \Theta :=& \frac{1}{n}(CD_0(CD_0)^{\mathsf{T}} + ED_0(ED_0)^{\mathsf{T}} + LD_0(CD_0)^{\mathsf{T}} + CD_0(LD_0)^{\mathsf{T}} \\ &+ LD_0(ED_0)^{\mathsf{T}} + ED_0(LD_0)^{\mathsf{T}} + CD_0(ED_0)^{\mathsf{T}} + CD_0(CD_0)^{\mathsf{T}}). \end{aligned} \tag{A.4}$$

Similarly to Kaslovsky and Meyer's notation, we define the terms in the eigendecomposition of $(1/n)XD_0(XD_0)^{\mathsf{T}}$ using

$$\frac{1}{n}XD_0(XD_0)^{\mathsf{T}} = \begin{bmatrix} \widehat{W}_1 & \widehat{W}_2 \end{bmatrix} \begin{bmatrix} \widehat{\Lambda}_1^W & 0 \\ 0 & \widehat{\Lambda}_2^W \end{bmatrix} \begin{bmatrix} \widehat{W}_1 & \widehat{W}_2 \end{bmatrix}^{\mathsf{T}}. \tag{A.5}$$

and

$$\frac{1}{n}LD_0(LD_0)^{\mathsf{T}} = \begin{bmatrix} W_1 & W_2 \end{bmatrix} \begin{bmatrix} \Lambda_1^W & 0 \\ 0 & \Lambda_2^W \end{bmatrix} \begin{bmatrix} W_1 & W_2 \end{bmatrix}^{\mathsf{T}}. \tag{A.6}$$

Note that we have replaced the U's with W's, Λ's with Λ^W's, and Δ with Θ to differentiate the Singer and Wu case from the authors' original analysis [**57**].

Our goal is to bound

$$\left\|\widehat{W}_1\widehat{W}_1^{\mathsf{T}} - W_1W_1^{\mathsf{T}}\right\|_F \tag{A.7}$$

by modifying Kaslovsky and Meyer's bounds and inserting them into Theorem A.1.1. Note that Theorem A.1.1 does not require any special structure of the input matrix (from which the eigendecomposition stems), e.g., this matrix does not need to have columns centered around $\mathbf{0}$.

As a last note, recall that Singer and Wu's local PCA uses the eigenvectors of $XD_0(XD_0)^{\mathsf{T}}$, not those of $(1/n)XD_0(XD_0)^{\mathsf{T}}$. However, since $XD_0(XD_0)^{\mathsf{T}} = LD_0(LD_0)^{\mathsf{T}} + n\Theta$, it suffices to simply multiply the relevant terms in Theorem A.1.1 by n in order to account for this difference. We do so after all the bounds are obtained.

A.4. Scaling Factor

In LPCA-SRC, we set the scaling factor $s := \|\boldsymbol{x}^{(n+1)} - \boldsymbol{x}_0\|^2$, where $\boldsymbol{x}^{(n+1)}$ is defined to be the farthest away of the $n+1$ samples (the $(n+1)$-nearest neighbors of $\boldsymbol{x}_0$ from the same class) from $\boldsymbol{x}_0$. Equivalently, $\boldsymbol{x}^{(n+1)}$ is the sample with the largest norm in the geometric setup, and thus it depends on all of the samples $\boldsymbol{x}^{(k)}$, $1 \le k \le n$. This choice of s is somewhat arbitrary; it provides a simple way of defining the neighborhood radius used in Singer and Wu's local PCA [**87**]. But all that is necessary in their algorithm is that s satisfy $s > \|\boldsymbol{x}^{(k)} - \boldsymbol{x}_0\|^2$ for $1 \le k \le n$. Further, without being able to assume that s is independent of X, it will be very difficult to amend Kaslovsky and Meyer's analysis to the Singer and Wu case. Thus we define s slightly differently here.

Treating $\boldsymbol{x}_0$ as the origin, we have that

$$\|\boldsymbol{x}^{(k)} - \mathbf{0}\|^2 = \|\boldsymbol{\ell}^{(k)}\|^2 + \|\boldsymbol{c}^{(k)}\|^2 + \|\boldsymbol{e}^{(k)}\|^2 + 2\left\langle \boldsymbol{\ell}^{(k)} + \boldsymbol{c}^{(k)}, \boldsymbol{e}^{(k)} \right\rangle$$

$$\leq r^2 + \frac{(K^{(+)})^2 r^4}{4} + \sum_{j=1}^{m} (e_i^{(k)})^2 + 2r \sum_{j=1}^{d} |e_j^{(k)}| + K^{(+)} r^2 \Big(\sum_{i=d+1}^{m} (e_i^{(k)})^2 \Big)^{1/2}$$

$$\leq \frac{(K^{(+)})^2 r^4}{4} + K^{(+)} r^2 \sqrt{m-d}\,(C\sigma) + r^2 + m(C\sigma)^2 + 2rd(C\sigma)$$

for all samples $\boldsymbol{x}^{(k)}$, $1 \leq k \leq n$. Here, $K^{(+)}$ and $K_i^{(+)}$ are Kaslovsky and Meyer's curvature constants defined using

$$K_i^{(+)} := \Big(\sum_{j=1}^{d} |\kappa_j^{(i)}|^2 \Big)^{1/2} \text{ and } K^{(+)} := \Big(\sum_{i=d+1}^{m} (K_i^{(+)})^2 \Big)^{1/2} \tag{A.8}$$

(we derive the bound $\|\boldsymbol{c}\|^2 \leq (K^{(+)})^2 r^4/4$ in Section A.5.4 below). Lastly, $C \in \mathbb{N}$ is chosen so that $|e_i^{(k)}| \leq \sigma$ with the desired probability. For example, setting $C = 3$ implies that the above bound holds with probability at least 99.7%.

Thus, in order to satisfy $s > \|\boldsymbol{x}^{(k)} - \boldsymbol{x}_0\|^2$, we set

$$s := \frac{(K^{(+)})^2 r^4}{4} + K^{(+)} r^2 \sqrt{m-d}(C\sigma) + r^2 + m(C\sigma)^2 + 2rd(C\sigma) + \epsilon, \tag{A.9}$$

for ϵ a small, positive constant.

REMARK A.4.1. *We briefly note that this bound on $\|\boldsymbol{x}^{(k)}\|^2$, $1 \leq k \leq n$, could be large, and so this assignment for the scaling factor s could offer little differentiation in the weights d_k (making them all close to 1) and rendering this important aspect of Singer and Wu's local PCA ineffective. At the end of our analysis, we consider what happens when a smaller value of s is used. See Section A.7.*

A.4.1. Breakdown of Terms. Following Kaslovsky and Meyer's lead, we consider the terms that need to be bounded. By Theorem A.1.1 (accounting for the change in notation), it is sufficient to bound the terms $\|W_1^{\mathsf{T}} \Theta\, W_1\|_F$, $\|W_2^{\mathsf{T}} \Theta\, W_2\|_F$, $\|W_2^{\mathsf{T}} \Theta\, W_1\|_F$, $\|W_1^{\mathsf{T}} \Theta\, W_2\|_F$, and λ_d^W, the dth largest eigenvalue of $(1/n) L D_0 (L D_0)^{\mathsf{T}}$, in order to bound Eq. (A.7). In particular, the bound on λ_d^W must be a lower bound.

Because Θ is symmetric (like Kaslovsky and Meyer's Δ), we have that

$$\|W_1^{\mathsf{T}} \Theta\, W_2\|_F = \|W_2^{\mathsf{T}} \Theta\, W_1\|_F,$$

and so we consider only the subscript pairs $(i,j) \in \{(1,1),(2,2),(2,1)\}$. Further, the equalities $W_1^\mathsf{T} C = 0$ and $W_2^\mathsf{T} L = 0$ follow from orthogonality, similarly to the standard local PCA case. Using these properties, the expansion of Θ using Eq. (A.4) and the triangle inequality, the terms that need bounding are

(1) λ_d^W,
(2) $\left\| W_1^\mathsf{T} \left(\frac{1}{n} L D_0 (E D_0)^\mathsf{T} \right) W_1 \right\|_F$,
(3) $\left\| W_1^\mathsf{T} \left(\frac{1}{n} E D_0 (E D_0)^\mathsf{T} \right) W_1 \right\|_F$,
(4) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} C D_0 (C D_0)^\mathsf{T} \right) W_2 \right\|_F$,
(5) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} C D_0 (E D_0)^\mathsf{T} \right) W_2 \right\|_F$,
(6) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} E D_0 (E D_0)^\mathsf{T} \right) W_2 \right\|_F$,
(7) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} C D_0 (L D_0)^\mathsf{T} \right) W_1 \right\|_F$,
(8) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} E D_0 (L D_0)^\mathsf{T} \right) W_1 \right\|_F$,
(9) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} C D_0 (E D_0)^\mathsf{T} \right) W_1 \right\|_F$,
(10) $\left\| W_2^\mathsf{T} \left(\frac{1}{n} E D_0 (E D_0)^\mathsf{T} \right) W_1 \right\|_F$.

Additionally, some of these Frobenius norm bounds will utilize upper bounds on the eigenvalues of the matrices $(1/n) L D_0 (L D_0)^\mathsf{T}$, $(1/n) C D_0 (C D_0)^\mathsf{T}$, and $(1/n) E D_0 (E D_0)^\mathsf{T}$. These are determined first.

REMARK A.4.2. *As implied in this detailed introduction, our derived tangent bound in the case of Singer and Wu's local PCA will be nearly entirely based on the derivation of Kaslovsky and Meyer's tangent bound [**57**]. For example, the terms listed above are exactly analogous to the terms they bound in their work. In nearly all instances, the methods used to obtain the bounds in the Singer and Wu case are exactly those used in Kaslovsky and Meyer's proofs, and they should be given full credit for these methods. To avoid redundancy, we will refrain from continuously writing "as the authors do..." or "analogously to Kaslovsky and Meyer..." and trust that the reader is well-aware that this is the case wherever applicable.*

*With this in mind, we will not "re-prove" Kaslovsky and Meyer's careful (and lengthy) analysis here. Instead, we primarily focus on the amendments needed to handle the Singer and Wu case. The reader is referred to Kaslovsky and Meyer's paper [**57**] for details and an instructive interpretation of their tangent bound.*

A.5. Bounding Terms

A.5.1. Linear Eigenvalues. We will determine both upper and lower bounds for the eigenvalues of $(1/n)LD_0(LD_0)^\mathsf{T}$ using the following theorem:

THEOREM A.5.1 (Tropp [**93**]). *Let $\{X_k\}$ be a finite sequence of independent, random, self-adjoint matrices satisfying $X_k \succeq 0$ and $\lambda_{\max}(X_k) \leq \lambda_\infty$ almost surely for all k, where $\lambda_{\max}(X_k)$ denotes the largest eigenvalue of X_k. Set*

$$\mu_{\min} := \lambda_{\min}\Big(\sum_{k=1}^{n} \mathbb{E}[X_k]\Big), \;\; \mu_{\max} := \lambda_{\max}\Big(\sum_{k=1}^{n} \mathbb{E}[X_k]\Big).$$

Then

$$\mathbb{P}\Big[\lambda_{\min}\Big(\sum_{k=1}^{n} X_k\Big) \leq (1-\delta)\mu_{\min}\Big] \leq d\Big[\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\Big]^{\frac{\mu_{\min}}{\lambda_\infty}}, \;\text{ for } \delta \in [0,1]$$

and

$$\mathbb{P}\Big[\lambda_{\max}\Big(\sum_{k=1}^{n} X_k\Big) \geq (1+\delta)\mu_{\max}\Big] \leq d\Big[\frac{e^{-\delta}}{(1+\delta)^{1+\delta}}\Big]^{\frac{\mu_{\max}}{\lambda_\infty}}, \;\text{ for } \delta \geq 0.$$

Noting that

$$\frac{1}{n}LD_0(LD_0)^\mathsf{T} = \frac{1}{n}\sum_{k=1}^{n} d_k\boldsymbol{\ell}^{(k)}(d_k\boldsymbol{\ell}^{(k)})^\mathsf{T} = \frac{1}{n}\sum_{k=1}^{n} d_k^2\boldsymbol{\ell}^{(k)}(\boldsymbol{\ell}^{(k)})^\mathsf{T},$$

we set $X_k := (1/n)\, d_k^2\boldsymbol{\ell}^{(k)}(\boldsymbol{\ell}^{(k)})^\mathsf{T}$. Clearly, X_k is self-adjoint and semi-positive definite. We need to determine the terms λ_∞, $\mu_{\min}$, and $\mu_{\max}$. Consider

$$\lambda_{\max}\Big(\frac{1}{n}d_k^2\boldsymbol{\ell}^{(k)}\boldsymbol{\ell}^{(k)\mathsf{T}}\Big) = \frac{1}{n}\lambda_{\max}(d_k^2\boldsymbol{\ell}^{(k)}\boldsymbol{\ell}^{(k)\mathsf{T}}) = \frac{1}{n}\|d_k\boldsymbol{\ell}^{(k)}\|_2^2 \leq \frac{1}{n}\|\boldsymbol{\ell}^{(k)}\|_2^2 \leq \frac{r^2}{n}.$$

Since this holds for all $1 \leq k \leq n$, we set $\lambda_\infty := r^2/n$.

Next, towards determining $\mu_{\min}$ and $\mu_{\max}$, we compute $\mathbb{E}[X_k]$. Since the distribution of the entries of X_k is the same for all k, we can drop the "k" notation, letting X, $\boldsymbol{\ell}$, and $\boldsymbol{x}$ denote the random variables from which the realizations $X^{(k)}$, $\boldsymbol{\ell}^{(k)}$, and $\boldsymbol{x}^{(k)}$, respectively, are drawn. Similarly,

we use ω to denote the random weight variable (from which d_k is drawn), i.e.,

$$\omega := \sqrt{1 - \frac{\|\boldsymbol{x}\|^2}{s}}. \tag{A.10}$$

It follows that we need to compute $\mathbb{E}[X] = (1/n)\mathbb{E}[\omega^2 \boldsymbol{\ell}\boldsymbol{\ell}^\mathsf{T}]$.

We will treat $\omega^2 \boldsymbol{\ell}\boldsymbol{\ell}^\mathsf{T} \in \mathbb{R}^{m\times m}$ as a $d \times d$ matrix, because all but the top-left $d \times d$ entries are zero. The expectation of the (l_1, l_2)-entry is given by

$$\begin{aligned}
\mathbb{E}[\omega^2 \ell_{l_1}\ell_{l_2}] &= \mathbb{E}\Big[\Big(1 - \frac{\|\boldsymbol{x}\|^2}{s}\Big)\ell_{l_1}\ell_{l_2}\Big] \\
&= \mathbb{E}[\ell_{l_1}\ell_{l_2}] - \frac{1}{s}\mathbb{E}\Big[\|\boldsymbol{x}\|^2 \ell_{l_1}\ell_{l_2}\Big] \\
&= \mathbb{E}[\ell_{l_1}\ell_{l_2}] - \frac{1}{s}\mathbb{E}\Big[\Big(\|\boldsymbol{\ell}\|^2 + \|\boldsymbol{c}\|^2 + \|\boldsymbol{e}\|^2 + 2\,\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\Big)\ell_{l_1}\ell_{l_2}\Big].
\end{aligned}$$

Since the vector formed from the first d entries of $\boldsymbol{\ell}$ is uniformly generated in $B_{\mathbf{0}}^d(r)$, we can write the jth coordinate of $\boldsymbol{\ell}$, $1 \le j \le d$, as

$$\ell_j = r u^{1/d} s_j, \tag{A.11}$$

where $u \sim \mathrm{unif}(0,1)$ and s_j is the jth coordinate of $\boldsymbol{s} \in \mathbb{R}^d$, a random vector uniformly sampled from the unit sphere S^{d-1} [**56**]. It follows that

$$\mathbb{E}[\ell_{l_1}\ell_{l_2}] = \begin{cases} \frac{r^2}{d}\mathbb{E}[u^{2/d}] = \frac{r^2}{d+2}, & l_1 = l_2 \\ 0, & l_1 \neq l_2. \end{cases}$$

It remains to compute $\mathbb{E}[\|\boldsymbol{\ell}\|^2 \ell_{l_1}\ell_{l_2}]$, $\mathbb{E}[\|\boldsymbol{c}\|^2 \ell_{l_1}\ell_{l_2}]$, $\mathbb{E}[\|\boldsymbol{e}\|^2 \ell_{l_1}\ell_{l_2}]$, and $\mathbb{E}[\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\, \ell_{l_1}\ell_{l_2}]$. Using the expansion in Eq. (A.11), we have

$$\mathbb{E}[\|\boldsymbol{\ell}\|^2 \ell_{l_1}\ell_{l_2}] = \frac{r^4 d}{d+4}\sum_{j=1}^{d} \mathbb{E}[s_{l_1} s_{l_2} s_j^2]$$

and

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{c}\|^2 \ell_{l_1}\ell_{l_2}] &= \mathbb{E}\Big[\Big(\frac{1}{4}\sum_{i=d+1}^{m}\Big(\sum_{j_1=1}^{d}\kappa_{j_1}^{(i)}\ell_{j_1}^2\Big)^2\Big)\ell_{l_1}\ell_{l_2}\Big] \\
&= \mathbb{E}\Big[\Big(\frac{r^4}{4}u^{4/d}\sum_{i=d+1}^{m}\Big(\sum_{j_1=1}^{d}\kappa_{j_1}^{(i)} s_{j_1}^2\Big)^2\Big) r^2 u^{2/d} s_{l_1} s_{l_2}\Big]
\end{aligned}$$

$$= \frac{r^6 d}{4(d+6)} \mathbb{E}\Big[\Big(\sum_{i=d+1}^{m} \Big(\sum_{j_1=1}^{d} \kappa_{j_1}^{(i)} s_{j_1}^2\Big)^2\Big) s_{l_1} s_{l_2}\Big]$$

$$= \frac{r^6 d}{4(d+6)} \Big[\sum_{i=d+1}^{m} \Big(\sum_{j_1=1}^{d} (\kappa_{j_1}^{(i)})^2 \mathbb{E}[s_{j_1}^4 s_{l_1} s_{l_2}] + \sum_{\substack{j_1, j_2 = 1 \\ j_1 \neq j_2}}^{d} \kappa_{j_1}^{(i)} \kappa_{j_2}^{(i)} \mathbb{E}[s_{j_1}^2 s_{j_2}^2 s_{l_1} s_{l_2}]\Big)\Big].$$

Since the distribution of $\boldsymbol{s} = [s_1, \ldots, s_d]^\mathsf{T}$ is invariant to sign changes in each coordinate, the expectation of any product of coordinates of $\boldsymbol{s}$ involving an odd power is 0. For example, for $l_1 \neq l_2$, we have that $\mathbb{E}[s_{l_1}^5 s_{l_2}] = \mathbb{E}[s_{l_1}^5 (-s_{l_2})] = -\mathbb{E}[s_{l_1}^5 s_{l_2}] \Rightarrow \mathbb{E}[s_{l_1}^5 s_{l_2}] = 0$. In the same manner, we obtain $\mathbb{E}[s_{j_1}^4 s_{l_1} s_{l_2}] = \mathbb{E}[s_{j_1}^2 s_{l_1} s_{l_2}] = \mathbb{E}[s_{j_1}^2 s_{j_2}^2 s_{l_1} s_{l_2}] = 0$ for $1 \leq l_1 \neq l_2 \leq d$. Thus these terms reduce to

$$\mathbb{E}[\|\boldsymbol{\ell}\|^2 \ell_{l_1}^2] = \frac{r^4 d}{d+4} \sum_{j_1=1}^{d} \mathbb{E}[s_{j_1}^2 s_{l_1}^2]$$

for arbitrary $1 \leq j_1 \leq d$, and

$$\mathbb{E}[\|\boldsymbol{c}\|^2 \ell_{l_1}^2] = \frac{r^6 d}{4(d+6)} \Big[\sum_{i=d+1}^{m} \Big(\sum_{j_1=1}^{d} (\kappa_{j_1}^{(i)})^2 \mathbb{E}[s_{j_1}^4 s_{l_1}^2] + \sum_{\substack{j_1, j_2 = 1 \\ j_1 \neq j_2}}^{d} \kappa_{j_1}^{(i)} \kappa_{j_2}^{(i)} \mathbb{E}[s_{j_1}^2 s_{j_2}^2 s_{l_1}^2]\Big)\Big]$$

for arbitrary $1 \leq j_1 \neq j_2 \leq d$.

For the remaining two terms in the expansion of $\mathbb{E}[\omega^2 \ell_{l_1} \ell_{l_2}]$, we have

$$\mathbb{E}[\|\boldsymbol{e}\|^2 \ell_{l_1} \ell_{l_2}] = \mathbb{E}[\|\boldsymbol{e}\|^2] \mathbb{E}[\ell_{l_1} \ell_{l_2}] = \frac{r^2 \sigma^2 m}{d+2}$$

for $l_1 = l_2$ by the independence of $\boldsymbol{\ell}$ and $\boldsymbol{e}$ (and equal to zero otherwise), and

$$\mathbb{E}[\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e} \rangle \, \ell_{l_1} \ell_{l_2}] = 0,$$

by the independence of $\boldsymbol{\ell} + \boldsymbol{c}$ and $\boldsymbol{e}$.

Putting everything together, we conclude that $\mathbb{E}[\omega^2 \ell_{l_1} \ell_{l_2}] = 0$ if $l_1 \neq l_2$ and that

$$\begin{aligned}\mathbb{E}[\omega^2 \ell_{l_1}^2] =& \frac{r^2}{d+2} - \frac{1}{s}\Big(\frac{r^2 \sigma^2 m}{d+2} + \frac{r^4 d}{d+4} \sum_{j_1=1}^{d} \mathbb{E}[s_{l_1}^2 s_{j_1}^2] \\ &+ \frac{r^6 d}{4(d+6)} \Big[\sum_{i=d+1}^{m} \Big(\sum_{j_1=1}^{d} (\kappa_{j_1}^{(i)})^2 \mathbb{E}[s_{j_1}^4 s_{l_1}^2] + \sum_{\substack{j_1, j_2 = 1 \\ j_1 \neq j_2}}^{d} \kappa_{j_1}^{(i)} \kappa_{j_2}^{(i)} \mathbb{E}[s_{j_1}^2 s_{j_2}^2 s_{l_1}^2]\Big)\Big]\Big)\end{aligned}$$

$$
\begin{aligned}
=&\frac{r^2}{d+2} - \frac{1}{s}\Bigg(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4 d}{d+4}\Big((d-1)\mathbb{E}[s_{l_1}^2 s_{j_1}^2] + \mathbb{E}[s_{l_1}^4]\Big) \\
&+ \frac{r^6 d}{4(d+6)}\Big[\Big((d-1)\mathbb{E}[s_{j_1}^4 s_{l_1}^2] + \mathbb{E}[s_{l_1}^6]\Big)(K^{(+)})^2 \\
&+ \Big(2(d-1)\mathbb{E}[s_{l_1}^4 s_{j_1}^2] + (d^2-3d+2)\mathbb{E}[s_{l_1}^2 s_{j_1}^2 s_{j_2}^2]\Big)\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\Bigg),
\end{aligned}
$$

where j_1 and j_2 are arbitrary in $1,\ldots,d$ and $j_1 \neq j_2 \neq l_1$. In the last term, we utilized Kaslovsky and Meyer's notation [**57**]

$$
K_{j_1 j_2}^{i_1 i_2} := \sum_{\substack{j_1, j_2 = 1 \\ j_1 \neq j_2}}^{d} \kappa_{j_1}^{(i_1)} \kappa_{j_2}^{(i_2)} \tag{A.12}
$$

(note that $K_{j_1 j_2}^{i_1 i_2}$ does not actually depend on j_1 and j_2). Also recall the definition of $K^{(+)}$ given in Eq. (A.8).

It follows that we can set

$$
\begin{aligned}
\lambda_{\min}\Big(\sum_{k=1}^{n}\mathbb{E}[X_k]\Big) &= \lambda_{\max}\Big(\sum_{k=1}^{n}\mathbb{E}[X_k]\Big) \\
&= \frac{r^2}{d+2} - \frac{1}{s}\Bigg(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4 d}{d+4}\Big((d-1)\mathbb{E}[s_{l_1}^2 s_{j_1}^2] + \mathbb{E}[s_{l_1}^4]\Big) \\
&\quad + \frac{r^6 d}{4(d+6)}\Big[\Big((d-1)\mathbb{E}[s_{j_1}^4 s_{l_1}^2] + \mathbb{E}[s_{l_1}^6]\Big)(K^{(+)})^2 \\
&\quad + \Big(2(d-1)\mathbb{E}[s_{l_1}^4 s_{j_1}^2] + (d^2-3d+2)\mathbb{E}[s_{l_1}^2 s_{j_1}^2 s_{j_2}^2]\Big)\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\Bigg) \\
&= \frac{r^2}{d+2} \\
&\quad - \frac{1}{s}\Bigg(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\Bigg) \\
&=: \mu,
\end{aligned}
$$

where $\mu = \mu_{\min} = \mu_{\max}$.

Let us set $\delta := \xi_{\lambda_d^W}\sqrt{\frac{2\lambda_\infty}{\mu}}$, for $\xi_{\lambda_d^W}$ a probability constant that relates to our bound on λ_d^W, the smallest (nonzero) eigenvalue of $(1/n)LD_0(LD_0)^{\mathsf{T}}$. By Theorem A.5.1, λ_d^W satisfies $\lambda_d^W > (1-\delta)\mu$

with probability greater than $1 - d\Big[\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}}$. Expanding terms renders

$$\lambda_d^W > (1-\delta)\mu = \left(1 - \xi_{\lambda_d^W}\frac{\sqrt{2\lambda_\infty}}{\sqrt{\mu}}\right)\mu = \mu - \xi_{\lambda_d^W}\sqrt{2\lambda_\infty}\sqrt{\mu},$$

and plugging in μ and λ_∞ produces

$$\begin{aligned}
\lambda_d^W > & \left[\frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)\right] \\
& - \xi_{\lambda_d^W}\sqrt{2\lambda_\infty}\sqrt{\frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)} \\
= & \frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right) \\
& - \xi_{\lambda_d^W}\frac{r^3\sqrt{2}}{\sqrt{n}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}.
\end{aligned}$$

Next, we show that this bound on λ_d^W holds with probability greater than $1 - de^{\xi^2_{\lambda_d^W}}$ by showing that

$$1 - d\Big[\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} \geq 1 - de^{\xi^2_{\lambda_d^W}}.$$

We have that

$$\begin{aligned}
1 - d\Big[\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} &\geq 1 - de^{-\xi^2_{\lambda_d^W}} \\
\Leftrightarrow \Big[\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} &\leq e^{-\xi^2_{\lambda_d^W}} \\
\Leftrightarrow \Big[\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} &\leq e^{-\frac{\delta^2}{2}\frac{\mu}{\lambda_\infty}} \\
\Leftrightarrow \frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}} &\leq e^{-\frac{\delta^2}{2}},
\end{aligned}$$

which is true for $\delta \in [0, 1]$. Note that this requires that

$$\text{(A.13)}\quad \xi_{\lambda_d^W} \leq \frac{\sqrt{n}}{\sqrt{2}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}.$$

We state this result formally:

LEMMA A.5.1. *For probability constant $\xi_{\lambda_d^W}$ obeying Eq.* (A.13)*, the smallest nonzero eigenvalue of $(1/n)LD_0(LD_0)^{\mathsf{T}}$ satisfies*

$$\lambda_d^W > \frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)$$
$$-\xi_{\lambda_d^W}\frac{r^3\sqrt{2}}{\sqrt{n}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}$$

with probability greater than $1 - de^{-\xi^2_{\lambda_d^W}}$, where $K^{(+)}$ is given by Eq. (A.8) *and $K^{ii}_{j_1 j_2}$ is given by Eq.* (A.12)*.*

Now we will produce an upper bound for the largest eigenvalue, λ_1^W, of $(1/n)LD_0(LD_0)^{\mathsf{T}}$. Set $\delta := \xi_{\lambda_1^W}(5/2)\sqrt{\lambda_\infty/\mu}$. Theorem A.5.1 states that λ_1^W satisfies $\lambda_1^W < (1+\delta)\mu$ with probability at least $1 - d\Big[\frac{e^{-\delta}}{(1+\delta)^{(1+\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}}$. Expanding terms gives

$$\begin{aligned}
\lambda_1^W <& (1+\delta)\mu \\
=& \Big(1 + \xi_{\lambda_1^W}\frac{5}{2}\frac{\sqrt{\lambda_\infty}}{\sqrt{\mu}}\Big)\mu \\
=& \mu + \xi_{\lambda_1^W}\frac{5}{2}\sqrt{\lambda_\infty}\sqrt{\mu} \\
=& \frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right) \\
&+ \xi_{\lambda_1^W}\frac{5r^3}{2\sqrt{n}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}.
\end{aligned}$$

Lastly, this bound on λ_1^W holds with probability greater than $1 - de^{-\xi_{\lambda_1^W}^2}$ since

$$\begin{aligned}
&1 - d\Big[\frac{e^{-\delta}}{(1+\delta)^{(1+\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} > 1 - de^{-\xi_{\lambda_1^W}^2} \\
&\Leftrightarrow \Big[\frac{e^{-\delta}}{(1+\delta)^{(1+\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} < e^{-\xi_{\lambda_1^W}^2} \\
&\Leftrightarrow \Big[\frac{e^{-\delta}}{(1+\delta)^{(1+\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} < e^{-\delta^2 \frac{4}{25} \frac{\mu}{\lambda_\infty^W}} \\
&\Leftrightarrow \Big[\frac{e^{-\delta}}{(1+\delta)^{(1+\delta)}}\Big]^{\frac{\mu}{\lambda_\infty}} < e^{-\frac{4}{25}\delta^2}
\end{aligned}$$

is true for all δ, $\xi_{\lambda_1} \geq 0$.

LEMMA A.5.2. *The largest eigenvalue of* $(1/n)LD_0(LD_0)^\mathsf{T}$ *satisfies*

$$\lambda_1^W < \frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\right)$$
$$+ \xi_{\lambda_1^W}\frac{5r^3}{2\sqrt{n}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\right)}$$

with probability greater than $1 - de^{-\xi_{\lambda_1^W}^2}$, *where* $K^{(+)}$ *is given by Eq.* (A.8) *and* $K_{j_1 j_2}^{ii}$ *is given by Eq.* (A.12).

We finish this section by setting

(A.14)
$$\lambda_{\text{bound}}^W(\xi) := \frac{r^2}{d+2} - \frac{1}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\right)$$
$$+ \xi\frac{5r^3}{2\sqrt{n}}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K_{j_1 j_2}^{ii}\Big]\right)}$$

for later use. By design, all the eigenvalues of $(1/n)LD_0(LD_0)^\mathsf{T}$ are no greater than $\lambda_{\text{bound}}^W(\xi)$ with probability depending on ξ.

A.5.2. Curvature Eigenvalue Bounds. We postpone this analysis until the end of Section A.5.4.

A.5.3. Noise Eigenvalue Bounds. To bound the eigenvalues of $W_i^{\mathsf{T}}(1/n)ED_0(ED_0)^{\mathsf{T}}W_i$ for $i = 1, 2$, we will use the following theorem:

THEOREM A.5.2 (Edelman [**35**], Vershynin [**97**]). *Let A be an $m \times n$ matrix, $m \leq n$, whose entries are independent standard normal random variables. Then for every $t \geq 0$, with probability at least $1 - 2\exp(-t^2/2)$, we have*

$$\sqrt{n} - \sqrt{m} - t \leq s_{\min}(A) \leq s_{\max}(A) \leq \sqrt{n} + \sqrt{m} + t,$$

where $s_{\min}(A)$ and $s_{\max}(A)$ denote the smallest and largest singular values of A, respectively.

In order to apply Theorem A.5.2, we first determine a constant α such that αED_0 has standard normal entries. This is a critical step in bounding all the terms involving the noise matrix E, as we will see.

Recall that $ED_0 = [d_1\boldsymbol{e}^{(1)}, \ldots, d_n\boldsymbol{e}^{(n)}] \in \mathbb{R}^{m\times n}$, where the entries of the vectors $\boldsymbol{e}^{(k)}$ have Gaussian distribution with mean 0 and variance σ^2. For $\boldsymbol{x}$ the random variable from which the samples $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ are realizations, let $\boldsymbol{e}$ denote its random noise component. As before, we define ω to be the random variable denoting the weight d_k from Eq. (A.10). Thus $\boldsymbol{x}^{(k)}$ and $d_k\boldsymbol{e}^{(k)}$ are the kth realizations of the random vectors $\boldsymbol{x}$ and $\omega\boldsymbol{e}$, respectively. Since an arbitrary coordinate e_l of $\boldsymbol{e}$ satisfies $e_l \sim \mathcal{N}(0, \sigma^2)$, the scaled noise component $\omega\boldsymbol{e}$ also has Gaussian-distributed entries with mean 0. Additionally, since ω and e_l are independent, we have that

$$\begin{aligned}\operatorname{var}(\alpha(\omega e_l)) = \alpha^2 \operatorname{var}(\omega e_l) &= \alpha^2(\mathbb{E}[(\omega e_l)^2] - \mathbb{E}[\omega e_l]^2) \\ &= \alpha^2(\mathbb{E}[\omega^2]\mathbb{E}[e_l^2] - \mathbb{E}[\omega]^2\mathbb{E}[e_l]^2) \\ &= \alpha^2\sigma^2\mathbb{E}[\omega^2].\end{aligned}$$

We compute

$$\begin{aligned}\mathbb{E}[\omega^2] = 1 - \frac{1}{s}\mathbb{E}[\|\boldsymbol{x}\|^2] &= 1 - \frac{1}{s}\Big(\mathbb{E}[\|\boldsymbol{\ell}\|^2] + \mathbb{E}[\|\boldsymbol{c}\|^2] + \mathbb{E}[\|\boldsymbol{e}\|^2] + 2\mathbb{E}[\langle\boldsymbol{\ell}, \boldsymbol{c}\rangle] + 2\mathbb{E}[\langle\boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle]\Big) \\ &= 1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big),\end{aligned}$$

where

$$\gamma := \frac{\sum_{i=d+1}^{m} \left(3K_{j_1 j_1}^{ii} + K_{j_1 j_2}^{ii}\right)}{2(d+2)(d+4)}. \tag{A.15}$$

This is a quantity defined in Kaslovsky and Meyer's original analysis [**57**]. It follows that setting

$$\alpha := \left(\sigma^2\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\right)^{-1/2} \tag{A.16}$$

will result in $\alpha E D_0$ having standard normal entries.

In order to bound the noise eigenvalues, we view the columns of E as n realizations of the random noise vector $\boldsymbol{e} \in \mathbb{R}^m$ and partition $\boldsymbol{e} = [\boldsymbol{e}_1^\mathsf{T}, \boldsymbol{e}_2^\mathsf{T}]^\mathsf{T}$ for $\boldsymbol{e}_1 \in \mathbb{R}^d$ and $\boldsymbol{e}_2 \in \mathbb{R}^{m-d}$. Then $W_i^\mathsf{T}(1/n)ED_0(ED_0)^\mathsf{T}W_i$ depends only on the realizations of $\boldsymbol{e}_1$ for $i = 1$ and those of $\boldsymbol{e}_2$ for $i = 2$. Next, define $Z := \alpha E D_0$ for α given by Eq. (A.16). By Theorem A.5.2 and standard inequalities, we have

$$\begin{aligned}
&\lambda_{\max}\Big(\frac{1}{n} W_1^\mathsf{T} Z Z^\mathsf{T} W_1\Big) \le \Big(1 - \frac{1}{n}\Big)^{-1}\Big(1 + \frac{5}{2\sqrt{n}}(\sqrt{d} + \xi_{\boldsymbol{e}_1}\sqrt{2})\Big) \\
&\Rightarrow \lambda_{\max}\Big(\frac{1}{n} W_1^\mathsf{T} E D_0 (E D_0)^\mathsf{T} W_1\Big) \le \frac{1}{\alpha^2}\Big(1 - \frac{1}{n}\Big)^{-1}\Big(1 + \frac{5}{2\sqrt{n}}(\sqrt{d} + \xi_{\boldsymbol{e}_1}\sqrt{2})\Big) \\
&\Rightarrow \lambda_{\max}\Big(\tfrac{1}{n} W_1^\mathsf{T} E D_0 (E D_0)^\mathsf{T} W_1\Big) \le \sigma^2\Big(1 - \tfrac{1}{s}\Big[\tfrac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big]\Big)\Big(1 - \tfrac{1}{n}\Big)^{-1}\Big(1 + \tfrac{5}{2\sqrt{n}}(\sqrt{d} + \xi_{\boldsymbol{e}_1}\sqrt{2})\Big)
\end{aligned}$$

with probability at least $1 - e^{-\xi_{\boldsymbol{e}_1}^2}$ for $n > 4(\sqrt{d} + \xi_{\boldsymbol{e}_1})$, and similarly

$$\lambda_{\max}\Big(\tfrac{1}{n} W_2^\mathsf{T} E D_0 (E D_0)^\mathsf{T} W_2\Big) \le \sigma^2\Big(1 - \tfrac{1}{s}\Big[\tfrac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big]\Big)\Big(1 - \tfrac{1}{n}\Big)^{-1}\Big(1 + \tfrac{5}{2\sqrt{n}}(\sqrt{m-d} + \xi_{\boldsymbol{e}_2}\sqrt{2})\Big)$$

with probability at least $1 - e^{-\xi_{\boldsymbol{e}_2}^2}$ for $n > 4(\sqrt{m-d} + \xi_{\boldsymbol{e}_2})$. We state these results formally:

LEMMA A.5.3. *The largest eigenvalue of* $W_1^\mathsf{T}(1/n)ED_0(ED_0)^\mathsf{T}W_1$ *satisfies*

$$\lambda_{\max}\Big(\tfrac{1}{n} W_1^\mathsf{T} E D_0 (E D_0)^\mathsf{T} W_1\Big) \le \sigma^2\Big(1 - \tfrac{1}{s}\Big[\tfrac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big]\Big)\Big(1 - \tfrac{1}{n}\Big)^{-1}\Big(1 + \tfrac{5}{2\sqrt{n}}(\sqrt{d} + \xi_{\boldsymbol{e}_1}\sqrt{2})\Big)$$

with probability at least $1 - 2e^{-\xi_{\boldsymbol{e}_1}^2}$ *for* $n > 4(\sqrt{d} + \xi_{\boldsymbol{e}_1})$. *The quantity* γ *is defined in Eq.* (A.15), *and the notation "*$\xi_{\boldsymbol{e}_1}$*" indicates that* $W_1^\mathsf{T} E D_0 (E D_0)^\mathsf{T} W_1$ *only depends on the first* d *coordinates of the realizations of the random noise vector* $\boldsymbol{e} \in \mathbb{R}^m$.

LEMMA A.5.4. *The largest eigenvalue of* $W_2^\mathsf{T}(1/n)ED_0(ED_0)^\mathsf{T}W_2$ *satisfies*

$$\lambda_{\max}\Big(\tfrac{1}{n}W_2^\mathsf{T}ED_0(ED_0)^\mathsf{T}W_2\Big) \leq \sigma^2\Big(1-\tfrac{1}{s}\Big[\tfrac{r^2d}{d+2}+\gamma r^4+\sigma^2 m\Big]\Big)\Big(1-\tfrac{1}{n}\Big)^{-1}\Big(1+\tfrac{5}{2\sqrt{n}}(\sqrt{m-d}+\xi_{\boldsymbol{e}_2}\sqrt{2})\Big)$$

with probability at least $1-2e^{-\xi_{\boldsymbol{e}_2}^2}$ *for* $n > 4(\sqrt{m-d}+\xi_{\boldsymbol{e}_2})$. *The quantity* γ *is defined in Eq.* (A.15), *and the notation "*$\xi_{\boldsymbol{e}_2}$*" indicates that* $W_2^\mathsf{T}ED_0(ED_0)^\mathsf{T}W_2$ *only depends on the last* $m-d$ *coordinates of the realizations of the random noise vector* $\boldsymbol{e} \in \mathbb{R}^m$.

A.5.4. Curvature Bounds. Our goal in this section is to bound

$$\big\|W_2^\mathsf{T}(1/n)CD_0(CD_0)^\mathsf{T}W_2\big\|_F.$$

Since W_2 contains the last $m-d$ eigenvectors of $(1/n)LD_0(LD_0)^\mathsf{T}$, we have that

$$W_2 = \begin{bmatrix} 0 & 0 \\ 0 & I_{(m-d)\times(m-d)} \end{bmatrix} \in \mathbb{R}^{m\times(m-d)},$$

because $LD_0(LD_0)^\mathsf{T}$ contains nonzeros only in its first d rows and columns. Further, since $CD_0(CD_0)^\mathsf{T}$ contains nonzeros only in its *last* $m-d$ rows and columns, it follows that $W_2^\mathsf{T}CD_0(CD_0)^\mathsf{T}W_2$ simply extracts the nonzero block of $CD_0(CD_0)^\mathsf{T}$. Consequently, $CD_0(CD_0)^\mathsf{T}$ and $W_2^\mathsf{T}CD_0(CD_0)^\mathsf{T}W_2$ have the same Frobenius norm.

We can view the quantity $(1/n)CD_0(CD_0)^\mathsf{T}$ as the empirical expectation of n samples of the scaled random curvature vector $\omega\boldsymbol{c}$, where ω again denotes the random weight variable given by Eq. (A.10). We write

$$\frac{1}{n}CD_0(CD_0)^\mathsf{T} = \hat{\mathbb{E}}[\omega^2\boldsymbol{c}\boldsymbol{c}^\mathsf{T}],$$

and we use the following theorem:

THEOREM A.5.3 (Shawe-Taylor and Cristianini [**84**]). *Given* n *realizations of a random matrix* Y *distributed with probability* $\mathbb{P}_Y$, *we have*

$$\mathbb{P}_Y\Big\{\big\|\mathbb{E}[Y]-\hat{\mathbb{E}}[Y]\big\|_F \leq \frac{R}{\sqrt{n}}(2+\xi\sqrt{2})\Big\} \geq 1-e^{-\xi^2},$$

where $R := \sup_{\mathrm{supp}(\mathbb{P}_Y)} \|Y\|_F$.

Since

$$\left| \left\| \mathbb{E}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2] \right\|_F - \|\hat{\mathbb{E}}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2]\|_F \right| \leq \left\| \mathbb{E}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2] - \hat{\mathbb{E}}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2] \right\|_F,$$

setting $Y := W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2$ in Theorem A.5.3 renders

$$\left| \left\| \mathbb{E}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2] \right\|_F - \left\| \hat{\mathbb{E}}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2] \right\|_F \right| \leq \frac{R_c}{\sqrt{n}}(2 + \xi\sqrt{2})$$

with probability at least $1 - e^{-\xi^2}$, where R_c is an upper bound on $\|W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2\|_F$ (note that it is unnecessary to use an exact supremum in Theorem A.5.3). Because $\|W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2\|_F \leq \|W_2^\mathsf{T} \omega \boldsymbol{c}\|_F \|\omega \boldsymbol{c}^\mathsf{T} W_2\|_F = \|W_2^\mathsf{T} \omega \boldsymbol{c}\|_F^2$, it suffices to bound this last quantity. Observe

$$\begin{aligned} \left\| W_2^\mathsf{T} \omega \boldsymbol{c} \right\|_F^2 = \|\omega \boldsymbol{c}\|^2 \leq \|\boldsymbol{c}\|^2 = \frac{1}{4} \sum_{i=d+1}^{m} \left(\kappa_1^{(i)} \ell_1^2 + \cdots + \kappa_d^{(i)} \ell_d^2 \right)^2 &\leq \frac{r^4}{4} \sum_{i=d+1}^{m} \left(\sum_{j=1}^{d} (\kappa_j^{(i)})^2 \right)^2 \\ &= \frac{r^4}{4} \sum_{i=d+1}^{m} (K_i^{(+)})^2 \\ &\leq \frac{(K^{(+)})^2 r^4}{4}, \end{aligned}$$

and so we set $R_c := \frac{K^{(+)} r^2}{2}$. The first inequality follows by Cauchy-Schwarz, and the terms $K_i^{(+)}$ and $K^{(+)}$ are the curvature constants defined in Eq. (A.8).

We next need to compute the true expectation $\mathbb{E}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{c}^\mathsf{T} W_2]$, which requires computing $\mathbb{E}[\omega^2 c_{l_1} c_{l_2}]$ for arbitrary $d+1 \leq l_1, l_2 \leq m$. We expand

$$\begin{aligned} \mathbb{E}[\omega^2 c_{l_1} c_{l_2}] &= \mathbb{E}\Big[\Big(1 - \frac{\|\boldsymbol{x}\|^2}{s} \Big) c_{l_1} c_{l_2} \Big] \\ &= \mathbb{E}[c_{l_1} c_{l_2}] - \mathbb{E}\Big[\frac{\|\boldsymbol{x}\|^2}{s} c_{l_1} c_{l_2} \Big]. \end{aligned}$$

From Kaslovsky and Meyer's original analysis [**57**], we have that

$$\mathbb{E}[c_{l_1} c_{l_2}] = \frac{[3K_{j_1 j_1}^{l_1 l_2} + K_{j_1 j_2}^{l_1 l_2}] r^4}{4(d+2)(d+4)},$$

where $K^{l_1 l_2}_{j_1 j_2}$ is given by Eq. (A.12) and

$$K^{l_1 l_2}_{j_1 j_1} := \sum_{j=1}^{d} \kappa^{(l_1)}_{j_1} \kappa^{(l_2)}_{j_1}. \tag{A.17}$$

Expanding the second term, we have

$$\begin{aligned}\mathbb{E}\Big[\frac{\|\boldsymbol{x}\|^2}{s} c_{l_1} c_{l_2}\Big] &= \frac{1}{s}\mathbb{E}\Big[\big(\|\boldsymbol{\ell}\|^2 + \|\boldsymbol{c}\|^2 + \|\boldsymbol{e}\|^2 + 2\,\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\,\big) c_{l_1} c_{l_2}\Big] \\ &= \frac{1}{s}\Big(\mathbb{E}[\|\boldsymbol{\ell}\|^2 c_{l_1} c_{l_2}] + \mathbb{E}[\|\boldsymbol{c}\|^2 c_{l_1} c_{l_2}] + \mathbb{E}[\|\boldsymbol{e}\|^2 c_{l_1} c_{l_2}] + 2\mathbb{E}[\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\, c_{l_1} c_{l_2}]\Big).\end{aligned}$$

We compute these quantities one at a time. For the first, we have

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{\ell}\|^2 c_{l_1} c_{j_1}] =& \mathbb{E}\Big[\Big(\sum_{j_1=1}^{d} \ell^2_{j_1}\Big)\Big(\sum_{j_2=1}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_2}\ell^4_{j_2} + \sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_3}\ell^2_{j_2}\ell^2_{j_3}\Big)\Big] \\ =& \mathbb{E}\Big[\sum_{j_1=1}^{d}\sum_{j_2=1}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_2}\ell^2_{j_1}\ell^4_{j_2}\Big] + \mathbb{E}\Big[\sum_{j_1=1}^{d}\sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_3}\ell^2_{j_1}\ell^2_{j_2}\ell^2_{j_3}\Big] \\ =& \frac{dr^6}{d+6}\left(\sum_{j_1=1}^{d}\sum_{j_2=1}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_2}\mathbb{E}[s^2_{j_1}s^4_{j_2}] + \sum_{j_1=1}^{d}\sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(j)}_{j_3}\mathbb{E}[s^2_{j_1}s^2_{j_2}s^2_{j_3}]\right) \\ =& \frac{dr^6}{d+6}\Big(\Big[d\,\mathbb{E}[s^6_{j_1}] + (d^2-d)\mathbb{E}[s^2_{j_1}s^4_{j_2}]\Big]K^{l_1 l_2}_{j_1 j_1} \\ &+ \Big[2d(d-1)\mathbb{E}[s^4_{j_1}s^2_{j_2}] + (d-2)(d^2-d)\mathbb{E}[s^2_{j_1}s^2_{j_2}s^2_{j_3}]\Big]K^{l_1 l_2}_{j_2 j_3}\Big),\end{aligned}$$

for $1 \le j_1 \neq j_2 \neq j_3 \le d$. Plugging in the expectations, this reduces to

$$\mathbb{E}[\|\boldsymbol{\ell}\|^2 c_{l_1} c_{j_1}] = \frac{dr^6 K^{l_1 l_2}_{j_1 j_1}}{d+6}.$$

For the curvature term, we have

$$\mathbb{E}[\|\boldsymbol{c}\|^2 c_{l_1} c_{l_2}] = \frac{1}{16}\mathbb{E}\Big[\Big(\sum_{i=d+1}^{m}\Big[\sum_{j_1=1}^{d}\kappa^{(i)}_{j_1}\ell^2_{j_1}\Big]^2\Big)\Big(\sum_{j_2=1}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_2}\ell^4_{j_2} + \sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d} \kappa^{(l_1)}_{j_2}\kappa^{(l_2)}_{j_3}\ell^2_{j_2}\ell^2_{j_3}\Big)\Big]$$

$$
\begin{aligned}
=&\frac{1}{16}\mathbb{E}\Big[\Big(\sum_{i=d+1}^{m}\sum_{j_1=1}^{d}(\kappa_{j_1}^{(i)})^2\ell_{j_1}^4+\sum_{i=d+1}^{m}\sum_{\substack{j_4,j_5=1\\ j_4\neq j_5}}^{d}\kappa_{j_4}^{(i)}\kappa_{j_5}^{(i)}\ell_{j_4}^2\ell_{j_5}^2\Big)\\
&\times\Big(\sum_{j_2=1}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_2}^{(l_2)}\ell_{j_2}^4+\sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_3}^{(l_2)}\ell_{j_2}^2\ell_{l_3}^2\Big)\Big]\\
=&\tfrac{1}{16}\mathbb{E}\Big[\sum_{i=d+1}^{m}\sum_{j_1=1}^{d}(\kappa_{j_1}^{(i)})^2\ell_{j_1}^4\sum_{j_2=1}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_2}^{(l_2)}\ell_{j_2}^4+\sum_{i=d+1}^{m}\sum_{\substack{j_4,j_5=1\\ j_4\neq j_5}}^{d}\kappa_{j_4}^{(i)}\kappa_{j_5}^{(i)}\ell_{j_4}^2\ell_{j_5}^2\sum_{j_2=1}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_2}^{(l_2)}\ell_{j_2}^4\\
&+\sum_{i=d+1}^{m}\sum_{j_1=1}^{d}(\kappa_{j_1}^{(i)})^2\ell_{j_1}^4\sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_3}^{(l_2)}\ell_{j_2}^2\ell_{j_3}^2+\sum_{i=d+1}^{m}\sum_{\substack{j_4,j_5=1\\ j_4\neq j_5}}^{d}\kappa_{j_4}^{(i)}\kappa_{j_5}^{(i)}\ell_{j_4}^2\ell_{j_5}^2\sum_{\substack{j_2,j_3=1\\ j_2\neq j_3}}^{d}\kappa_{j_2}^{(l_1)}\kappa_{j_3}^{(l_2)}\ell_{j_2}^2\ell_{j_3}^2\Big],
\end{aligned}
$$

which further reduces to

$$
\begin{aligned}
\mathbb{E}[\|\boldsymbol{c}\|^2c_{l_1}c_{l_2}]=&\frac{dr^8}{16(d+8)}\Bigg(\Big[d\,\mathbb{E}[s_{j_1}^8]+(d^2-d)\mathbb{E}[s_{j_1}^4s_{j_2}^4]\Big](K^{(+)})^2K_{j_1j_1}^{l_1l_2}\\
&+\Big[2d(d-1)\mathbb{E}[s_{j_1}^6s_{j_2}^2]+(d-2)(d^2-d)\mathbb{E}[s_{j_1}^4s_{j_2}^2s_{j_3}^2]\Big]K_{j_1j_1}^{l_1l_2}\sum_{i=d+1}^{m}K_{j_1j_2}^{ii}\\
&+\Big[2d(d-1)\mathbb{E}[s_{j_1}^6s_{j_2}^2]+(d-2)(d^2-d)\mathbb{E}[s_{j_1}^4s_{j_2}^2s_{j_3}^2]\Big](K^{(+)})^2K_{j_1j_2}^{l_1l_2}\\
&+\Big[2d(d-1)\mathbb{E}[s_{j_1}^4s_{j_2}^4]+(d^2-d)(d-2)(d-3)\mathbb{E}[s_{j_1}^2s_{j_2}^2s_{j_3}^2s_{j_4}^2]\\
&+((d^2-d)^2-2d(d-1)-(d^2-d)(d-2)(d-3))\mathbb{E}[s_{j_1}^4s_{j_2}^2s_{j_3}^2]\Big]K_{j_1j_2}^{l_1l_2}\sum_{i=d+1}^{m}K_{j_1j_2}^{ii}\Bigg),
\end{aligned}
$$

where $1\leq j_1\neq j_2\neq j_3\neq j_4\leq d$ (recall that the subscript indexing for the curvature constants is arbitrary). Plugging in the expectations renders

$$
\mathbb{E}[\|\boldsymbol{c}\|^2c_{l_1}c_{l_2}]=\frac{dr^8K_{j_1j_1}^{l_1l_2}}{16(d+2)(d+8)}\Big(3(K^{(+)})^2+\frac{(d-1)(5d^2-10d+48)}{(d+4)(d+6)}\sum_{i=d+1}^{m}K_{j_1j_2}^{ii}\Big).
$$

For the noise term, we have

$$
\mathbb{E}[\|\boldsymbol{e}\|^2c_{l_1}c_{l_2}]=\mathbb{E}[\|\boldsymbol{e}\|^2]\mathbb{E}[c_{l_1}c_{l_2}]=d\sigma^2\frac{[3K_{j_1j_1}^{l_1l_2}+K_{j_1j_2}^{l_1l_2}]r^4}{4(d+2)(d+4)}
$$

for $1\leq j_1\neq j_2\leq d$, and finally, for the last term we have

$$\begin{aligned}\mathbb{E}[\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\, c_{l_1} c_{l_2}] &= \mathbb{E}\Big[\Big(\sum_{j=1}^{d} \ell_j e_j + \sum_{i=d+1}^{m} c_i e_i\Big) c_{l_1} c_{l_2}\Big] \\ &= \sum_{j=1}^{d} \mathbb{E}[\ell_j e_j c_{l_1} c_{l_2}] + \sum_{i=d+1}^{m} \mathbb{E}[c_i e_i c_{l_1} c_{l_2}] \\ &= \sum_{j=1}^{d} \mathbb{E}[e_j]\mathbb{E}[\ell_j c_{l_1} c_{l_2}] + \sum_{i=d+1}^{m} \mathbb{E}[e_i]\mathbb{E}[c_i c_{l_1} c_{l_2}] \\ &= 0.\end{aligned}$$

Considering all the terms together, the true expectation satisfies

$$\begin{aligned}\mathbb{E}[\omega^2 c_{l_1} c_{l_2}] =& \mathbb{E}[c_{l_1} c_{l_2}] - \frac{1}{s}\mathbb{E}[\|\boldsymbol{\ell}\|^2 c_{l_1} c_{l_2}] + \mathbb{E}[\|\boldsymbol{c}\|^2 c_{l_1} c_{l_2}] + \mathbb{E}[\|\boldsymbol{e}\|^2 c_{l_1} c_{l_2}] + 2\mathbb{E}[\langle \boldsymbol{\ell} + \boldsymbol{c}, \boldsymbol{e}\rangle\, c_{l_1} c_{l_2}] \\ =& \Big(1 - \frac{d\sigma^2}{s}\Big) \frac{[3K^{l_1 l_2}_{j_1 j_1} + K^{l_1 l_2}_{j_1 j_2}] r^4}{4(d+2)(d+4)} \\ & - \frac{d r^6 K^{l_1 l_2}_{j_1 j_1}}{s} \left[\frac{1}{d+6} + \frac{r^2}{16(d+2)(d+8)} \left(3(K^{(+)})^2 + \frac{(d-1)(5d^2 - 10d + 48)}{(d+4)(d+6)} \sum_{i=d+1}^{m} K^{ii}_{j_1 j_2} \right) \right].\end{aligned}$$

Theorem A.5.3 implies that

$$\Big\| W_2^{\mathsf{T}} \Big(\frac{1}{n} C D_0 (C D_0)^{\mathsf{T}} \Big) W_2 \Big\|_F \leq \Big(\sum_{i,j=d+1}^{m} (\mathbb{E}[\omega^2 c_i c_j])^2 \Big)^{1/2} + \frac{K^{(+)}(2 + \xi\sqrt{2})}{2\sqrt{n}}$$

with probability at least $1 - e^{-\xi^2}$. We set $\xi = \xi_{\boldsymbol{c}}$ to identify the probability constant corresponding to this curvature bound. Note that the above inequality also holds with probability *strictly* greater than $1 - 2e^{-\xi_{\boldsymbol{c}}^2}$. Formally, we have

Lemma A.5.5. *The Frobenius norm of the curvature term satisfies*

$$\Big\| W_2^{\mathsf{T}} \Big(\frac{1}{n} C D_0 (C D_0)^{\mathsf{T}} \Big) W_2 \Big\|_F \leq \Big(\sum_{l_1, l_2 = d+1}^{m} (\mathbb{E}[\omega^2 c_{l_1} c_{l_2}])^2 \Big)^{1/2} + \frac{K^{(+)}(2 + \xi_{\boldsymbol{c}}\sqrt{2})}{2\sqrt{n}}$$

with probability greater than $1 - 2e^{-\xi_c^2}$*, where*

$$\begin{aligned}\mathbb{E}[\omega^2 c_{l_1} c_{l_2}] =& \Big(1 - \frac{d\sigma^2}{s}\Big) \frac{[3K^{l_1 l_2}_{j_1 j_1} + K^{l_1 l_2}_{j_1 j_2}] r^4}{4(d+2)(d+4)} - \frac{dr^6 K^{l_1 l_2}_{j_1 j_1}}{s} \Bigg[\frac{1}{d+6} \\ &+ \frac{r^2}{16(d+2)(d+8)} \Big(3(K^{(+)})^2 + \frac{(d-1)(5d^2 - 10d + 48)}{(d+4)(d+6)} \sum_{i=d+1}^{m} K^{ii}_{j_1 j_2} \Big) \Bigg].\end{aligned} \tag{A.18}$$

Here, $K^{(+)}$*,* $K^{ii}_{j_1 j_2}$*, and* $K^{l_1 l_2}_{j_1 j_1}$ *are defined in Eq.* (A.8)*, Eq.* (A.12)*, and Eq.* (A.17)*, respectively.*

To obtain a bound on γ_1^W, the largest eigenvalue of $(1/n)CD_0(CD_0)^\mathsf{T}$, we use that

$$\begin{aligned}\gamma_1^W &= \Big\| \frac{1}{n} CD_0 (CD_0)^\mathsf{T} \Big\|_2 \\ &\leq \Big\| \frac{1}{n} CD_0 (CD_0)^\mathsf{T} \Big\|_F \\ &= \Big\| W_2^\mathsf{T} \Big(\frac{1}{n} CD_0 (CD_0)^\mathsf{T} \Big) W_2 \Big\|_F \\ &\leq \Big(\sum_{i,j=d+1}^{m} (\mathbb{E}[\omega^2 c_i c_j])^2 \Big)^{1/2} + \frac{K^{(+)}(2 + \xi\sqrt{2})}{2\sqrt{n}}\end{aligned}$$

with probability greater than $1 - 2e^{-\xi^2}$. This produces the following definition:

$$\gamma^W_{\text{bound}}(\xi) := \Big(\sum_{i,j=d+1}^{m} (\mathbb{E}[\omega^2 c_i c_j])^2 \Big)^{1/2} + \frac{K^{(+)}(2 + \xi\sqrt{2})}{2\sqrt{n}}. \tag{A.19}$$

A.5.5. Curvature-Linear Bounds. Next we bound $\|W_2^\mathsf{T}(1/n)CD_0(LD_0)^\mathsf{T}W_1\|_F$ using the same approach as in the previous section. We observe that

$$\frac{1}{n} W_2^\mathsf{T} CD_0 (LD_0)^\mathsf{T} W_1 = \hat{\mathbb{E}}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T} W_1]$$

and use Theorem A.5.3 to obtain

$$\Big| \big\| \mathbb{E}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T} W_1] \big\|_F - \big\| \hat{\mathbb{E}}[W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T} W_1] \big\|_F \Big| \leq \frac{R}{\sqrt{n}} (2 + \xi\sqrt{2})$$

with probability at least $1 - e^{-\xi^2}$ and $R \geq \sup \|W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T} W_1\|_F$. Since

$$\big\| W_2^\mathsf{T} \omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T} W_1 \big\|_F \leq \big\| W_2^\mathsf{T} \omega \boldsymbol{c} \big\|_F \big\| \omega \boldsymbol{\ell}^\mathsf{T} W_1 \big\|_F \leq \big\| W_2^\mathsf{T} \boldsymbol{c} \|_F \| \boldsymbol{\ell}^\mathsf{T} W_1 \big\|_F,$$

we can use R_c from the previous section and set $R_\ell := r$. Thus we obtain $R := R_c R_\ell$.

To complete the bound, we will show that $\|\mathbb{E}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1]\|_F = 0$. Let us first consider $\mathbb{E}[\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}]$. Since the matrix $\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}$ has nonzero entries $\omega^2 c_{l_1}\ell_{l_2}$ for $l_1 = d{+}1,\ldots,m$ and $l_2 = 1,\ldots,d$, we have that

$$\begin{aligned}\omega^2 c_{l_1}\ell_{l_2} &= \frac{1}{2}\omega^2\Big(\kappa_1^{(l_1)}\ell_1^2 + \cdots + \kappa_d^{(l_1)}\ell_d^2\Big)\ell_{l_2}\\ &= \frac{1}{2}\omega^2\Big(\kappa_1^{(l_1)}\ell_1^2\ell_{j_1} + \cdots + \kappa_j^{(l_1)}\ell_j^3 + \cdots + \kappa_d^{(l_1)}\ell_d^2\ell_{l_2}\Big).\end{aligned}$$

It follows that

$$\mathbb{E}[\omega^2 c_{l_1}\ell_{l_2}] = \frac{1}{2}\Big(\kappa_1^{(l_1)}\mathbb{E}[\omega^2\ell_1^2\ell_{l_2}] + \cdots + \kappa_{j_1}^{(l_1)}\mathbb{E}[\omega^2\ell_{l_2}^3] + \cdots + \kappa_d^{(l_1)}\mathbb{E}[\omega^2\ell_d^2\ell_{l_2}]\Big). \tag{A.20}$$

Expanding the first type of expectation in Eq. (A.20) renders

$$\begin{aligned}\mathbb{E}[\omega^2\ell_{j_2}^2\ell_{l_2}] &= \mathbb{E}\Big[\Big(1 - \frac{\|\boldsymbol{x}\|^2}{s}\Big)(r^2u^{2/d}s_{j_2}^2)(ru^{1/d}s_{l_2})]\\ &= \frac{dr^3}{d+3}\Big(\mathbb{E}[s_{j_2}^2 s_{l_2}] - \frac{1}{s}\mathbb{E}[\|\boldsymbol{x}\|^2 s_{j_2}^2 s_{l_2}]\Big)\\ &= \frac{-dr^3}{s(d+3)}\mathbb{E}[\|\boldsymbol{x}\|^2 s_{j_2}^2 s_{l_2}]\\ &= \frac{-dr^3}{s(d+3)}\Big(\mathbb{E}[\|\boldsymbol{\ell}\|^2 s_{j_2}^2 s_{l_2}] + \mathbb{E}[\|\boldsymbol{c}\|^2 s_{j_2}^2 s_{l_2}] + \mathbb{E}[\|\boldsymbol{e}\|^2 s_{j_2}^2 s_{l_2}] + 2\mathbb{E}[\langle\boldsymbol{\ell}+\boldsymbol{c},\boldsymbol{e}\rangle\, s_{j_2}^2 s_{l_2}]\Big),\end{aligned}$$

for $1 \le j_2 \ne l_2 \le d$. By independence of the noise term, this reduces to

$$\mathbb{E}[\omega^2\ell_{j_2}^2\ell_{l_2}] = \frac{-dr^3}{s(d+3)}\Big(\mathbb{E}[\|\boldsymbol{\ell}\|^2 s_{j_2}^2 s_{l_2}] + \mathbb{E}[\|\boldsymbol{c}\|^2 s_{j_2}^2 s_{l_2}]\Big).$$

However, the remaining terms in this expression are also zero, because $\mathbb{E}[s_{j_3}^2 s_{j_2}^2 s_{l_2}] = \mathbb{E}[s_{j_3}^4 s_{j_2}^2 s_{l_2}]$ $= \mathbb{E}[s_{j_3}^2 s_{j_4}^2 s_{j_2}^2 s_{l_2}] = 0$.

For the second type of term in Eq. (A.20), we have

$$\begin{aligned}\mathbb{E}[\omega^2\ell_{l_1}^3] &= \frac{dr^3}{d+3}\Big(\mathbb{E}[s_{l_1}^3] - \frac{1}{s}\mathbb{E}[\|\boldsymbol{x}\|^2 s_{l_1}^3]\Big)\\ &= \frac{-dr^3}{s(d+3)}\Big(\mathbb{E}[\|\boldsymbol{\ell}\|^2 s_{l_1}^3] + \mathbb{E}[\|\boldsymbol{c}\|^2 s_{l_1}^3]\Big) \quad \text{(by independence of the noise terms)}\\ &= 0,\end{aligned}$$

because $\mathbb{E}[s_{j_1}^2 s_{l_1}^3] = \mathbb{E}[s_{j_1}^4 s_{l_1}^3] = \mathbb{E}[s_{j_1}^2 s_{j_2}^2 s_{l_1}^3] = 0$. Thus we conclude that $\mathbb{E}[\omega^2 \boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}] = 0$. Lastly, because the multiplication of $\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}$ by W_2^T and W_1 preserves its entries' zero expectation, it follows that $\mathbb{E}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1] = 0$.

Thus we have that

$$\begin{aligned}\left\|\hat{\mathbb{E}}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1]\right\|_F &\leq \left\|\hat{\mathbb{E}}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1 - \mathbb{E}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1\right\|_F + \left\|\mathbb{E}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1\right\|_F \\ &= \left\|\hat{\mathbb{E}}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1 - \mathbb{E}[W_2^\mathsf{T}\omega^2\boldsymbol{c}\boldsymbol{\ell}^\mathsf{T}W_1\right\|_F \\ &\leq \frac{R_c R_\ell}{\sqrt{n}}(2+\xi\sqrt{2})\end{aligned}$$

with probability at least $1 - e^{-\xi^2}$. We label the corresponding probability constant $\xi =: \xi_{\boldsymbol{c}\boldsymbol{l}}$. Plugging in R_c and R_ℓ, we obtain

LEMMA A.5.6. *The Frobenius norm of the curvature-linear term satisfies*

$$\left\|W_2^\mathsf{T}\Big(\frac{1}{n}CD_0(LD_0)^\mathsf{T}\Big)W_1\right\|_F \leq \frac{K^{(+)}r^3}{2\sqrt{n}}(2+\xi_{\boldsymbol{c}\boldsymbol{l}}\sqrt{2})$$

with probability at least $1 - e^{-\xi_{\boldsymbol{c}\boldsymbol{l}}^2}$. *The quantity* $K^{(+)}$ *is defined in Eq.* (A.8).

A.5.6. Noise Bounds. To bound the Frobenius norm of the matrix $W_i^\mathsf{T}(1/n)ED_0(ED_0)^\mathsf{T}W_j$ where $(i,j) \in \{(1,1),(2,2),(2,1)\}$, recall that the entries of $Z := \alpha ED_0$ (for α in Eq. (A.16)) are standard normal. Fortunately for our purposes, Kaslovsky and Meyer's original analysis for the analogous noise bound (in the standard local PCA case) is completely distribution-based, i.e., it depends only on the positions of nonzeros and the distribution of the matrix entries. Thus the results for the Singer and Wu case follow directly from making the necessary substitutions in Kaslovsky and Meyer's derived bounds, in particular, adjusting them to our definition of α. Again, see their paper [**57**] for the derivation.

LEMMA A.5.7. *The following noise term satisfies*

$$\begin{aligned}\left\|W_1^\mathsf{T}\Big(\frac{1}{n}ED_0(ED_0)^\mathsf{T}\Big)W_1\right\|_F \leq &\sqrt{d}\Big(\sigma^2\Big[1-\frac{1}{s}\Big(\frac{r^2 d}{d+2}+\gamma r^4+\sigma^2 m\Big)\Big]\Big)\Big(1-\frac{1}{n}\Big)^{-1} \\ &\times\Big[1+\frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{d}+\xi_{\boldsymbol{e}_1}\sqrt{2})\Big]\end{aligned}$$

with probability greater than $1-2e^{-\xi_{\boldsymbol{e}_1}^2}$. *The notation "*$\xi_{\boldsymbol{e}_1}$*" indicates that* $W_1^\mathsf{T} ED_0(ED_0)^\mathsf{T} W_1$ *only depends on the first* d *coordinates of the realizations of the random noise vector* $\boldsymbol{e} \in \mathbb{R}^m$.

LEMMA A.5.8. *The following noise term satisfies*

$$\Big\|W_2^\mathsf{T}\Big(\frac{1}{n}ED_0(ED_0)^\mathsf{T}\Big)W_2\Big\|_F \leq \sqrt{m-d}\Big(\sigma^2\Big[1-\frac{1}{s}\Big(\frac{r^2 d}{d+2}+\gamma r^4+\sigma^2 m\Big)\Big]\Big)\Big(1-\frac{1}{n}\Big)^{-1} \times \Big[1+\frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{m-d}+\xi_{\boldsymbol{e}_2}\sqrt{2})\Big]$$

with probability greater than $1-2e^{-\xi_{\boldsymbol{e}_2}^2}$. *The notation "*$\xi_{\boldsymbol{e}_2}$*" indicates that* $W_2^\mathsf{T} ED_0(ED_0)^\mathsf{T} W_2$ *only depends on the last* $m-d$ *coordinates of the realizations of the random noise vector* $\boldsymbol{e} \in \mathbb{R}^m$.

LEMMA A.5.9. *The following noise term satisfies*

$$\Big\|W_2^\mathsf{T}\Big(\frac{1}{n}ED_0(ED_0)^\mathsf{T}\Big)W_1\Big\|_F \leq \frac{\sqrt{d(m-d)}}{\sqrt{n}}\Big(\sigma^2\Big[1-\frac{1}{s}\Big(\frac{r^2 d}{d+2}+\gamma r^4+\sigma^2 m\Big)\Big]\Big)\Big(1-\frac{1}{n}\Big)^{-1} \times \Big(1+\frac{\sqrt{m-d}+\xi_{\boldsymbol{e}_2}\sqrt{2}}{\sqrt{n}}\Big)\Big[1+\frac{6}{5}\frac{\xi_{\boldsymbol{e}_3}}{\sqrt{d(m-d)}}\Big]$$

with probability greater than $1-2e^{-\xi_{\boldsymbol{e}_2}^2}-e^{-\xi_{\boldsymbol{e}_3}^2}$. *Here,* $\xi_{\boldsymbol{e}_3}$ *is a probability constant that satisfies* $\xi_{\boldsymbol{e}_3} \leq 0.7\sqrt{d(m-d)}$.

A.5.7. Linear-Noise Bounds. To bound $\|W_i^\mathsf{T}(1/n)ED_0(LD_0)^\mathsf{T}W_1\|_F$ for $i \in \{1,2\}$, define the matrices $P_1 \in \mathbb{R}^{m\times d}$ and $Q_1 \in \mathbb{R}^{d\times d}$ so that $P_1Q_1 = W_1$ with

$$P_1 = \begin{bmatrix} I_{d\times d} \\ 0_{(m-d)\times d} \end{bmatrix}.$$

Then it follows that Q_1 is an orthogonal matrix, and we can write

$$\Big\|W_1^\mathsf{T}\Big(\frac{1}{n}ED_0(LD_0)^\mathsf{T}\Big)U_1\Big\|_F = \frac{1}{n}\|(P_1Q_1)^\mathsf{T}ED_0(LD_0)^\mathsf{T}(P_1Q_1)\|_F = \frac{1}{n}\|P_1^\mathsf{T}ED_0(LD_0)^\mathsf{T}P_1\|_F.$$

In order to simplify the term in the Frobenius norm, we expand $P_1^\mathsf{T}LD_0$ in terms of its singular value decomposition:

$$P_1^\mathsf{T}LD_0 = Q\Sigma V^\mathsf{T}$$

$$\Rightarrow \|P_1^\mathsf{T}\Big(\frac{1}{n}ED_0(LD_0)^\mathsf{T}\Big)P_1\|_F = \frac{1}{n}\|P_1^\mathsf{T}ED_0(Q\Sigma V^\mathsf{T})^\mathsf{T}\|_F$$

$$= \frac{1}{n}\|P_1^\mathsf{T} E D_0 \, V\Sigma Q^\mathsf{T}\|_F$$
$$= \frac{1}{n}\|P_1^\mathsf{T} E D_0 \, V\Sigma\|_F.$$

Now, Σ contains the singular values of $P_1^\mathsf{T} L D_0$, which are equal to the square roots of the eigenvalues of $P_1^\mathsf{T} L D_0 (P_1^\mathsf{T} L D_0)^\mathsf{T}$. Since P_1 is just a resizing matrix, these are equal to the square roots of the eigenvalues of $L D_0 (L D_0)^\mathsf{T}$. We can write

$$\begin{aligned}
\Big\| P_1^\mathsf{T}\Big(\frac{1}{n} E (L D_0)^\mathsf{T}\Big) P_1 \Big\|_F &= \frac{1}{n}\|P_1^\mathsf{T} E \, V\Sigma\|_F \\
&\le \frac{1}{n}\|P_1^\mathsf{T} E \, V\|_F \sqrt{\lambda_{\max}(L D_0 (L D_0)^\mathsf{T})} \\
&\le \frac{1}{n}\|P_1^\mathsf{T} E \, V\|_F \sqrt{n\lambda_{\max}((1/n) L D_0 (L D_0)^\mathsf{T})} \\
&\le \frac{\sqrt{\lambda_1^W}}{\sqrt{n}}\|P_1^\mathsf{T} E \, V\|_F,
\end{aligned}$$

where $\lambda_1^W := \lambda_{\max}((1/n) L D_0 (L D_0)^\mathsf{T})$, as before.

Despite the matrix V's dependence on the realization of the matrix $L D_0$, the steps to bound $\|P_1^\mathsf{T} E \, V\|_F$ follow identically to Kaslovsky and Meyer's analysis in the standard local PCA case. This is (again) a consequence of the standard normal distribution of the entries of $\alpha E D_0$ (for α in Eq. (A.16)), as well as the authors' approach of first bounding the (analogous) quantity for a fixed V followed by the removal of this conditioning.

We obtain the following bound:

LEMMA A.5.10. *The following noise-linear term satisfies*

$$\Big\| W_1^\mathsf{T}\Big(\frac{1}{n} E D_0 (L D_0)^\mathsf{T}\Big) W_1 \Big\|_F \le \sigma \sqrt{1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)} \, \frac{\sqrt{\lambda^W_{\text{bound}}(\xi_{\lambda_1^W})}}{\sqrt{n}} \Big(d + \frac{6}{5}\xi_{e\ell}\Big)$$

with probability greater than $(1 - e^{-\xi_{e\ell}^2/2})(1 - d e^{-\xi_{\lambda_1^W}^2})$. *Here,* $\xi_{e\ell}$ *and* $\xi_{\lambda_1^W}$ *are probability constants, and* λ^W_{bound} *is defined in Eq.* (A.14).

The case that $i = 2$ follows similarly:

LEMMA A.5.11. *The following noise-linear term satisfies*

$$\left\|W_2^{\mathsf{T}}\Big(\frac{1}{n}ED_0(LD_0)^{\mathsf{T}}\Big)W_1\right\|_F \leq \sigma\sqrt{1-\frac{1}{s}\Big(\frac{r^2d}{d+2}+\gamma r^4+\sigma^2 m\Big)}\frac{\sqrt{\lambda^W_{\text{bound}}(\xi_{\lambda_1^W})}}{\sqrt{n}}\Big(\sqrt{d(m-d)}+\frac{6}{5}\xi_{\boldsymbol{e\ell}}\Big)$$

with probability greater than $(1-e^{-\xi^2_{\boldsymbol{e\ell}}/2})(1-de^{-\xi^2_{\lambda_1^W}})$. *Here,* $\xi_{\boldsymbol{e\ell}}$ *and* $\xi_{\lambda_1^W}$ *are probability constants, and* λ^W_{bound} *is defined in Eq.* (A.14).

A.5.8. Curvature-Noise Bounds. Analogous to the original theorem, the bounds for $\|W_2^{\mathsf{T}}(1/n)CD_0(ED_0)^{\mathsf{T}}W_j\|_F$, $j=1,2$, can be obtained in a similar manner to the linear-noise bounds in Section A.5.7. We obtain the following lemmas:

LEMMA A.5.12. *The following curvature-noise term satisfies*

$$\left\|W_2^{\mathsf{T}}\Big(\frac{1}{n}CD_0(ED_0)^{\mathsf{T}}\Big)W_1\right\|_F \leq \sigma\sqrt{1-\frac{1}{s}\Big(\frac{r^2d}{d+2}+\gamma r^4+\sigma^2 m\Big)}\frac{\sqrt{\gamma^W_{\text{bound}}(\xi_{\boldsymbol{c}})}}{\sqrt{n}}\Big(\sqrt{d(m-d)}+\frac{6}{5}\xi_{\boldsymbol{ce}}\Big)$$

with probability greater than $(1-e^{-\xi^2_{\boldsymbol{ce}}})(1-2e^{-\xi^2_{\boldsymbol{c}}})$. *Here,* $\xi_{\boldsymbol{ce}}$ *and* $\xi_{\boldsymbol{c}}$ *are probability constants, and* γ^W_{bound} *is defined in Eq.* (A.19).

LEMMA A.5.13. *The following curvature-noise term satisfies*

$$\left\|W_2^{\mathsf{T}}\Big(\frac{1}{n}CD_0(ED_0)^{\mathsf{T}}\Big)W_2\right\|_F \leq \sigma\sqrt{1-\frac{1}{s}\Big(\frac{r^2d}{d+2}+\gamma r^4+\sigma^2 m\Big)}\frac{\sqrt{\gamma^W_{\text{bound}}(\xi_{\boldsymbol{c}})}}{\sqrt{n}}\Big(m-d+\frac{6}{5}\xi_{\boldsymbol{ce}}\Big)$$

with probability greater than $(1-e^{-\xi^2_{\boldsymbol{ce}}})(1-2e^{-\xi^2_{\boldsymbol{c}}})$. *Here,* $\xi_{\boldsymbol{ce}}$ *and* $\xi_{\boldsymbol{c}}$ *are probability constants, and* γ^W_{bound} *is defined in Eq.* (A.19).

A.6. End Result

A.6.1. Summary of Computed Bounds. Let us consider the bounds computed in the previous section. We set $\xi_{\lambda_d^W}=\xi_{\lambda_1^W}=:\xi_{\lambda^W}$ and

$$\xi_{\boldsymbol{c\ell}}=\xi_{\boldsymbol{e\ell}}=\xi_{\boldsymbol{ce}}=\xi_{\boldsymbol{e}_1}=\xi_{\boldsymbol{e}_2}=\xi_{\boldsymbol{e}_3}=\xi_{\boldsymbol{c}}=:\xi.$$

We also multiply each bound by n (to remove the extra $1/n$ factor), making the eigenvalue notation substitutions $n\lambda_d^W=:\lambda_d$, $n\lambda^W_{\text{bound}}=:\lambda_{\text{bound}}$, and $n\gamma^W_{\text{bound}}:=\gamma_{\text{bound}}$ for convenience. This produces

the equations

(A.21)

$$\lambda_{\text{bound}}(\xi) := \frac{nr^2}{d+2} - \frac{n}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right) + \xi\frac{5r^3\sqrt{n}}{2}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}$$

and

$$\gamma_{\text{bound}}(\xi) := n\Big(\sum_{i,j=d+1}^{m}(\mathbb{E}[\omega^2 c_i c_j])^2\Big)^{1/2} + \frac{\sqrt{n}K^{(+)}(2+\xi\sqrt{2})}{2}, \tag{A.22}$$

where again, $\mathbb{E}[\omega^2 c_i c_j]$ is defined in Eq. (A.18). It makes sense to then set the probability constant $\xi_{\lambda^W} =: \xi_\lambda$.

This renders the following list of bounds:

(1) $\lambda_d > \frac{nr^2}{d+2} - \frac{n}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right) - \xi_\lambda r^3\sqrt{2n}\sqrt{\frac{1}{d+2} - \frac{1}{s}\left(\frac{\sigma^2 m}{d+2} + \frac{r^2}{d+4} + \frac{r^4}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)}$ with probability greater than $1 - de^{-\xi_\lambda^2}$ and for ξ_λ satisfying Eq. (A.13) (by Lemma A.5.1),

(2) $\big\|W_2^{\mathsf{T}} C D_0 (C D_0)^{\mathsf{T}} W_2\big\|_F \leq n\Big(\sum_{i,j=d+1}^{m}(\mathbb{E}[\omega^2 c_i c_j])^2\Big)^{1/2} + \frac{K^{(+)}\sqrt{n}(2+\xi\sqrt{2})}{2} = \gamma_{\text{bound}}(\xi)$ with probability greater than $1 - 2e^{-\xi^2}$, where $\mathbb{E}[\omega^2 c_i c_j]$ is given by Eq. (A.18) (by Lemma A.5.5),

(3) $\big\|W_2^{\mathsf{T}} C D_0 (L D_0)^{\mathsf{T}} W_1\big\|_F \leq \frac{K^{(+)}\sqrt{n}r^3}{2}(2+\xi\sqrt{2})$ with probability at least $1 - e^{-\xi^2}$ (by Lemma A.5.6),

(4) $\big\|W_1^{\mathsf{T}} E D_0 (E D_0)^{\mathsf{T}} W_1\big\|_F \leq \frac{n^2\sigma^2\sqrt{d}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big[1 + \frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{d} + \xi\sqrt{2})\Big]$ with probability greater than $1 - 2e^{-\xi^2}$ (by Lemma A.5.7),

(5) $\big\|W_2^{\mathsf{T}} E D_0 (E D_0)^{\mathsf{T}} W_2\big\|_F \leq \frac{n^2\sigma^2\sqrt{m-d}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big[1 + \frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{m-d} + \xi\sqrt{2})\Big]$ with probability greater than $1 - 2e^{-\xi^2}$ (by Lemma A.5.8),

(6) $\big\|W_2^{\mathsf{T}} E D_0 (E D_0)^{\mathsf{T}} W_1\big\|_F \leq \frac{n^{3/2}\sigma^2\sqrt{d(m-d)}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big(1 + \frac{\sqrt{m-d}+\xi\sqrt{2}}{\sqrt{n}}\Big)\Big[1 + \frac{6}{5}\frac{\xi}{\sqrt{d(m-d)}}\Big]$ with probability greater than $1 - 2e^{-\xi^2} - e^{-\xi^2}$ and for $\xi \leq 0.7\sqrt{d(m-d)}$ (by Lemma A.5.9),

(7) $\left\|W_1^{\mathsf{T}} E D_0 (L D_0)^{\mathsf{T}} W_1\right\|_F \leq \sigma \sqrt{1 - \frac{1}{s}\left(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\right)} \sqrt{\lambda_{\text{bound}}(\xi_\lambda)} \left(d + \frac{6}{5}\xi\right)$ with probability greater than $(1 - e^{-\xi^2/2})(1 - d e^{-\xi_\lambda^2})$ (by Lemma A.5.10),

(8) $\left\|W_2^{\mathsf{T}} E D_0 (L D_0)^{\mathsf{T}} W_1\right\|_F \leq \sigma \sqrt{1 - \frac{1}{s}\left(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\right)} \sqrt{\lambda_{\text{bound}}(\xi_\lambda)} \left(\sqrt{d(m-d)} + \frac{6}{5}\xi\right)$ with probability greater than $(1 - e^{-\xi^2/2})(1 - d e^{-\xi_\lambda^2})$ (by Lemma A.5.11),

(9) $\left\|W_2^{\mathsf{T}} C D_0 (E D_0)^{\mathsf{T}} W_1\right\|_F \leq \sigma \sqrt{1 - \frac{1}{s}\left(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\right)} \sqrt{\gamma_{\text{bound}}(\xi)} \left(\sqrt{d(m-d)} + \frac{6}{5}\xi\right)$ with probability greater than $(1 - e^{-\xi^2})(1 - 2e^{-\xi^2})$ (by Lemma A.5.12),

(10) $\left\|W_2^{\mathsf{T}} C D_0 (E D_0)^{\mathsf{T}} W_2\right\|_F \leq \sigma \sqrt{1 - \frac{1}{s}\left(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\right)} \sqrt{\gamma_{\text{bound}}(\xi)} \left(m - d + \frac{6}{5}\xi\right)$ (by Lemma A.5.13).

A.6.2. Putting It All Together. We restate Theorem A.1.1 with our notation for easy reference:

THEOREM A.6.1 (Restated theorem: Davis and Kahan [**25**], Stewart [**89**]). *Set*

$$\delta := \lambda_d - \left\|W_1^{\mathsf{T}}(n\Theta)W_1\right\|_F - \left\|W_2^{\mathsf{T}}(n\Theta)W_2\right\|_F.$$

Define the following conditions:

- *Condition 1:* $\delta > 0$,
- *Condition 2:* $\left\|W_1^{\mathsf{T}}(n\Theta)W_2\right\|_F \left\|W_2^{\mathsf{T}}(n\Theta)W_1\right\|_F < \frac{1}{4}\delta^2$.

If both conditions hold, then

$$\left\|WW^{\mathsf{T}} - \widehat{W}\widehat{W}^{\mathsf{T}}\right\|_F \leq 2\sqrt{2}\,\frac{\left\|W_2^{\mathsf{T}}(n\Theta)W_1\right\|_F}{\delta}.$$

From the triangle inequality and the equalities $W_1^{\mathsf{T}} C = 0$ and $W_2^{\mathsf{T}} L = 0$, we have

$$\begin{aligned}
\left\|W_1^{\mathsf{T}}(n\Theta)W_1\right\|_F \leq{}& 2\left\|W_1^{\mathsf{T}} L D_0 (E D_0) W_1\right\|_F + \left\|W_1^{\mathsf{T}} E D_0 (E D_0) W_1\right\|_F, \\
\left\|W_2^{\mathsf{T}}(n\Theta)W_2\right\|_F \leq{}& \left\|W_2^{\mathsf{T}} C D_0 (C D_0) W_2\right\|_F + 2\left\|W_2^{\mathsf{T}} C D_0 (E D_0) W_2\right\|_F + \left\|W_2^{\mathsf{T}} E D_0 (E D_0) W_2\right\|_F, \\
\left\|W_2^{\mathsf{T}}(n\Theta)W_1\right\|_F \leq{}& \left\|W_2^{\mathsf{T}} C D_0 (L D_0) W_1\right\|_F + \left\|W_2^{\mathsf{T}} E D_0 (L D_0) W_1\right\|_F \\
&+ \left\|W_2^{\mathsf{T}} C D_0 (E D_0) W_1\right\|_F + \left\|W_2^{\mathsf{T}} E D_0 (E D_0) W_1\right\|_F.
\end{aligned}$$

It now is simply a matter of plugging the computed bounds into Theorem A.6.1.

THEOREM A.6.2 (Tangent bound, modified for Singer and Wu's method of local PCA). *Let s be the scaling factor defined in Eq.* (A.9)*, and let ξ_λ satisfying Eq.* (A.13) *and $\xi \leq 0.7\sqrt{d(m-d)}$ be probability constants with $n > 4(\max(\sqrt{d}, \sqrt{m-d}) + \xi)$. Assume the setup defined above, in particular: (i) the eigendecompositions in Eq.* (A.5) *and Eq.* (A.6)*; (ii) the curvature constants defined in Eq.* (A.8)*, Eq.* (A.12)*, Eq.* (A.15)*, and Eq.* (A.17)*; and (iii) the eigenvalue bounds in Eq.* (A.21) *and Eq.* (A.22) *(the latter of which uses Eq.* (A.18)*). Further define*

$$
\begin{aligned}
\delta :=& \frac{nr^2}{d+2} - \frac{n}{s}\left(\frac{r^2\sigma^2 m}{d+2} + \frac{r^4}{d+4} + \frac{r^6}{4(d+6)}\Big[\frac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \frac{d-1}{d+2}\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right) \\
& - \xi_\lambda r^3\sqrt{2n}\sqrt{\tfrac{1}{d+2} - \tfrac{1}{s}\left(\tfrac{\sigma^2 m}{d+2} + \tfrac{r^2}{d+4} + \tfrac{r^4}{4(d+6)}\Big[\tfrac{3(d+4)}{(d+2)(d+4)}(K^{(+)})^2 + \tfrac{d-1}{d+2}\textstyle\sum_{i=d+1}^{m} K^{ii}_{j_1 j_2}\Big]\right)} \\
& - 2\sigma\sqrt{1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)}\sqrt{\lambda_{\text{bound}}(\xi_\lambda)}\Big(d + \frac{6}{5}\xi\Big) \\
& - \frac{n^2\sigma^2\sqrt{d}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big[1 + \frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{d} + \xi\sqrt{2})\Big] \\
& - \gamma_{\text{bound}}(\xi) - 2\sigma\sqrt{1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)}\sqrt{\gamma_{\text{bound}}(\xi)}\Big(m - d + \frac{6}{5}\xi\Big) \\
& - \frac{n^2\sigma^2\sqrt{m-d}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big[1 + \frac{5}{2}\frac{1}{\sqrt{n}}(\sqrt{m-d} + \xi\sqrt{2})\Big],
\end{aligned}
$$

and

$$
\begin{aligned}
\beta :=& \frac{K^{(+)}\sqrt{n}r^3}{2}(2 + \xi\sqrt{2}) + \sigma\sqrt{1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)}\sqrt{\lambda_{\text{bound}}(\xi_\lambda)}\Big(\sqrt{d(m-d)} + \frac{6}{5}\xi\Big) \\
& + \sigma\sqrt{1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)}\sqrt{\gamma_{\text{bound}}(\xi)}\Big(\sqrt{d(m-d)} + \frac{6}{5}\xi\Big) \\
& + \frac{n^{3/2}\sigma^2\sqrt{d(m-d)}}{n-1}\Big[1 - \frac{1}{s}\Big(\frac{r^2 d}{d+2} + \gamma r^4 + \sigma^2 m\Big)\Big]\Big(1 + \frac{\sqrt{m-d} + \xi\sqrt{2}}{\sqrt{n}}\Big)\Big[1 + \frac{6}{5}\frac{\xi}{\sqrt{d(m-d)}}\Big].
\end{aligned}
$$

If the two conditions

- $\delta > 0$, *and*
- $\beta < \frac{1}{2}\delta$

hold, then the angle between the true and estimated tangent plane, computed using Singer and Wu's local PCA algorithm, satisfies

$$\left\|WW^{\mathsf{T}} - \widehat{W}\widehat{W}^{\mathsf{T}}\right\|_F \leq \frac{2\sqrt{2}\,\beta}{\delta} \tag{A.23}$$

with probability greater than $1 - 2de^{-\xi_\lambda^2} - 9e^{-\xi^2}$.

A.7. Adjusting the Scaling Factor

As a final consideration, we investigate how the bound in Theorem A.6.2 is affected when a smaller scaling factor s is used, i.e., when we use a tighter upper bound on $\|\boldsymbol{x}\|^2$ than the one in Eq. (A.9) (recall that this is the case in the LPCA-SRC algorithm). By the quantities in Theorem A.6.2, making s smaller will cause β and the subtracted terms in the expression for δ to decrease. However, the first term in the expression for δ (the lower bound on λ_d) also decreases when s is made smaller, and so the overall behavior of the tangent bound is not immediately obvious.

We expect that, in general, the tangent bound will decrease as s decreases to $\max_{\boldsymbol{x}\in\mathcal{X}} \|\boldsymbol{x}\|_2^2 + \epsilon$, for a fixed set of neighbor samples $\mathcal{X}$ and a small but non-negligible constant $\epsilon > 0$. In the case that $d = 1$ (as in our tangent vector distance theorem in Chapter 6), δ can be written informally as

$$\begin{aligned}\delta \approx n\Bigg[& r^2\Big[\frac{1}{3} - \frac{1}{s}\Big(\frac{\sigma^2 m}{3} + \frac{1}{5} + \frac{3r^2\|\boldsymbol{\kappa}\|_2^2}{84}\Big)\Big] - r^4\|\boldsymbol{\kappa}\|_2\left(\frac{1}{20} - \frac{1}{s}\Big(\frac{\sigma^2}{20} + \frac{r^2}{7} + \frac{r^2\|\boldsymbol{\kappa}\|_2^2}{144}\Big)\right) \\ & - \sigma^2\Big[1 - \frac{1}{s}\Big(\frac{r^2}{3} + \frac{r^4\|\boldsymbol{\kappa}\|_2^2}{10} + \sigma^2 m\Big)\Big](1 + \sqrt{m-1})\Bigg],\end{aligned}$$

where $\boldsymbol{\kappa} := [\kappa_1^{(2)}, \ldots, \kappa_1^{(m)}]^{\mathsf{T}} \in \mathbb{R}^{m-1}$. Thus we can (heuristically) confirm that the bound in Eq. (A.23) decreases as s decreases by checking to see if

$$\begin{aligned}\frac{1}{n}\delta'(s) \approx\ & r^2\Big[\frac{1}{3} + \frac{1}{s^2}\Big(\frac{\sigma^2 m}{3} + \frac{1}{5} + \frac{3r^2\|\boldsymbol{\kappa}\|_2^2}{84}\Big)\Big] - r^4\|\boldsymbol{\kappa}\|_2\left(\frac{1}{20} + \frac{1}{s^2}\Big(\frac{\sigma^2}{20} + \frac{r^2}{7} + \frac{r^2\|\boldsymbol{\kappa}\|_2^2}{144}\Big)\right) \\ & - \sigma^2\Big[1 + \frac{1}{s^2}\Big(\frac{r^2}{3} + \frac{r^4\|\boldsymbol{\kappa}\|_2^2}{10} + \sigma^2 m\Big)\Big](1 + \sqrt{m-1})\end{aligned}$$

is non-negative. However, we note that this is not sufficient to reject this hypothesis, and that doing so would require showing that

$$g'(s) := \frac{\partial}{\partial s}\Big(\frac{\beta(s)}{\delta(s)}\Big)$$

is less than 0. Unfortunately, the expansion of $g'(s)$, even in the case that $d = 1$, is quite complicated.

Bibliography

[1] Asif, M., and Romberg, J. ℓ_1 homotopy: A MATLAB toolbox for homotopy algorithms in ℓ_1-norm minimization problems. `http://users.ece.gatech.edu/~sasif/homotopy/`, accessed 31.3.2015. 2009–2013.

[2] AT&T Laboratories Cambridge. The database of faces. `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`, accessed 26.3.2016. 1992-1994.

[3] Belhumeur, P. N., and Kriegman, D. J. What is the set of images of an object under all possible lighting conditions? In *1996 IEEE Conference on Computer Vision and Pattern Recognition* (June 1996), pp. 270–277.

[4] Belkin, M., and Niyogi, P. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computing 15*, 6 (2003), 1373–1396.

[5] Bentley, J. L. Multidimensional binary search trees used for associative searching. *Commun. ACM 18*, 9 (1975), 509–517.

[6] Beyreuther, M., Carniel, R., and Wassermann, J. Automatic earthquake detection and classification with continuous hidden Markov models: a possible tool for monitoring Las Candas caldera in Tenerife. *IOP Conference Series: Earth and Environmental Science 3*, 1 (2008), 12–28.

[7] Boser, B. E., Guyon, I. M., and Vapnik, V. N. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (1992), COLT '92, ACM, pp. 144–152.

[8] Bruckstein, A. M., Donoho, D. L., and Elad, M. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Rev. 51*, 1 (2009), 34–81.

[9] Cai, T. T., Wang, L., and Xu, G. New bounds for restricted isometry constants. *IEEE Trans. Inform. Theory 56*, 9 (2010), 4388–4394.

[10] Candès, E. J., and Plan, Y. Near-ideal model selection by ℓ_1 minimization. *Ann. Statist. 37*, 5A (2009), 2145–2177.

[11] Candès, E. J., and Plan, Y. A probabilistic and RIPless theory of compressed sensing. *IEEE Trans. Inform. Theory 57*, 11 (2011), 7235–7254.

[12] Candès, E. J., Romberg, J., and Tao, T. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory 52*, 2 (2006), 489–509.

[13] Candès, E. J., Romberg, J. K., and Tao, T. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math. 59*, 8 (2006), 1207–1223.

[14] Candès, E. J., and Tao, T. Decoding by linear programming. *IEEE Trans. Inform. Theory 51*, 12 (2005), 4203–4215.

[15] Candès, E. J., and Tao, T. Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans. Inform. Theory 52*, 12 (2006), 5406–5425.

[16] Ceruti, C., Bassis, S., Rozza, A., Lombardi, G., Casiraghi, E., and Campadelli, P. DANCo: An intrinsic dimensionality estimator exploiting angle and norm concentration. *Pattern Recognition 47*, 8 (2014), 2569 – 2581.

[17] Cevikalp, H., Yavuz, H. S., Cay, M. A., and Barkana, A. Two-dimensional subspace classifiers for face recognition. *Neurocomputing 72*, 46 (2009), 1111 – 1120.

[18] Chang, J.-M., and Kirby, M. Face recognition under varying viewing conditions with subspace distance. In *International Conference on Artificial Intelligence and Pattern Recognition (AIPR-09)* (2009), pp. 16–23.

[19] Chang, J.-M., and Pacheco, J. I. Classifying handwritten digits on the Grassmann manifold. In *Proceedings of The International Conference on Image Processing, Computer Vision, & Pattern Recognition* (2011), vol. 1, pp. 36–41.

[20] Chang, Y., Hu, C., and Turk, M. Manifold of facial expression. In *Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures* (2003), vol. 24, pp. 605–614.

[21] Cheng, B., Yang, J., Yan, S., Fu, Y., and Huang, T. S. Learning with $l1$-graph for image analysis. *IEEE Trans. Image Process. 19*, 4 (2010), 858–866.

[22] Cortes, C., and Vapnik, V. Support-vector networks. *Machine Learning 20*, 3 (1995), 273–297.

[23] Cover, T., and Hart, P. Nearest neighbor pattern classification. *IEEE Trans. Inform. Theory 13*, 1 (1967), 21–27.

[24] Dalal, N., and Triggs, B. Histograms of oriented gradients for human detection. In *2005 IEEE Conference on Computer Vision and Pattern Recognition* (June 2005), vol. 1, pp. 886–893.

[25] Davis, C., and Kahan, W. M. The rotation of eigenvectors by a perturbation. III. *SIAM J. Numer. Anal. 7* (1970), 1–46.

[26] Deng, W., Hu, J., and Guo, J. Extended SRC: Undersampled face recognition via intraclass variant dictionary. *IEEE Trans. Pattern Anal. Mach. Intell. 34*, 9 (2012), 1864–1870.

[27] Donoho, D. L. Neighborly polytopes and sparse solutions of underdetermined linear equations. Tech. rep., 2005.

[28] Donoho, D. L. Compressed sensing. *IEEE Trans. Inform. Theory 52*, 4 (2006), 1289–1306.

[29] Donoho, D. L. For most large underdetermined systems of linear equations the minimal l_1-norm solution is also the sparsest solution. *Comm. Pure Appl. Math. 59*, 6 (2006), 797–829.

[30] Donoho, D. L., and Elad, M. Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization. *Proc. Natl. Acad. Sci. USA 100*, 5 (2003), 2197–2202.

[31] Donoho, D. L., Elad, M., and Temlyakov, V. N. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Trans. Inform. Theory 52*, 1 (2006), 6–18.

[32] Donoho, D. L., and Tanner, J. Sparse nonnegative solution of underdetermined linear equations by linear programming. *Proc. Natl. Acad. Sci. USA 102*, 27 (2005), 9446–9451.

[33] Donoho, D. L., and Tsaig, Y. Fast solution of l_1-norm minimization problems when the solution may be sparse. *IEEE Trans. Inform. Theory 54*, 11 (2008), 4789–4812.

[34] Duarte, M. F., Davenport, M. A., Takbar, D., Laska, J. N., Sun, T., Kelly, K. F., and Baraniuk, R. G. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine 25*, 2 (March 2008), 83–91.

[35] Edelman, A. Eigenvalues and condition numbers of random matrices. *SIAM J. Matrix Anal. Appl. 9*, 4 (1988), 543–560.

[36] Eldar, Y. C., Kuppinger, P., and Bölcskei, H. Block-sparse signals: uncertainty relations and efficient recovery. *IEEE Trans. Signal Process. 58*, 6 (2010), 3042–3054.

[37] Eldar, Y. C., and Mishali, M. Robust recovery of signals from a structured union of subspaces. *IEEE Trans. Inform. Theory 55*, 11 (2009), 5302–5316.

[38] Eldén, L. *Matrix methods in data mining and pattern recognition*, vol. 4 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2007.

[39] Elhamifar, E., and Vidal, R. Sparse subspace clustering. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (June 2009), pp. 2790–2797.

[40] Elhamifar, E., and Vidal, R. Robust classification using structured sparse representation. In *2011 IEEE Conference on Computer Vision and Pattern Recognition* (June 2011), pp. 1873–1879.

[41] Elhamifar, E., and Vidal, R. Block-sparse recovery via convex optimization. *IEEE Trans. Signal Process. 60*, 8 (2012), 4094–4107.

[42] Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of Eugenics 7*, 2 (1936), 179–188.

[43] Friedman, J., Hastie, T., and Tibshirani, R. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software 33*, 1 (2010), 1–22.

[44] Georghiades, A. S., Belhumeur, P. N., and Kriegman, D. J. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 6 (2001), 643–660.

[45] Giaquinta, M., and Modica, G. *Mathematical Analysis: An Introduction to Functions of Several Variables*. Birkhäuser Boston, Inc., Boston, MA, 2009. Translated and revised from the 2005 Italian original.

[46] Gribonval, R., and Nielsen, M. Sparse representations in unions of bases. *IEEE Trans. Inform. Theory 49*, 12 (2003), 3320–3325.

[47] Hand, D. J., and Henley, W. E. Statistical classification methods in consumer credit scoring: a review. *Journal of the Royal Statistical Society: Series A (Statistics in Society) 160*, 3 (1997), 523–541.

[48] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, second ed. Springer Series in Statistics. Springer, New York, 2009.

[49] Hastie, T., Tibshirani, R., and Wainwright, M. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press, Taylor & Francis, 2015.

[50] He, X., Yan, S., Hu, Y., Niyogi, P., and Zhang, H.-J. Face recognition using Laplacianfaces. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 3 (2005), 328–340.

[51] Herrmann, F. J., Friedlander, M. P., and Yilmaz, O. Fighting the curse of dimensionality: Compressive sensing in exploration seismology. *IEEE Signal Processing Magazine 29*, 3 (May 2012), 88–100.

[52] Ho, J., Xie, Y., and Vemuri, B. C. On a nonlinear generalization of sparse coding and dictionary learning. In *ICML (3)* (2013), vol. 28 of *JMLR Proceedings*, JMLR.org, pp. 1480–1488.

[53] Hui, K.-H., Li, C.-L., and Zhang, L. Sparse neighbor representation for classification. *Pattern Recognition Letters 33*, 5 (2012), 661 – 669.

[54] Hull, J. J. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell. 16*, 5 (1994), 550–554.

[55] Kang, C., Liao, S., Xiang, S., and Pan, C. Kernel sparse representation with pixel-level and region-level local feature kernels for face recognition. *Neurocomputing 133* (2014), 141 – 152.

[56] Kaslovsky, D. Matlab code for "Non-Asymptotic Analysis of Tangent Space Perturbation". `http://danielkaslovsky.com/code`, accessed 18.6.15. 2012.

[57] Kaslovsky, D. N., and Meyer, F. G. Non-asymptotic analysis of tangent space perturbation. *Inf. Inference 3*, 2 (2014), 134–187.

[58] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (Nov 1998), 2278–2324.

[59] Lee, K.-C., Ho, J., and Kriegman, D. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 5 (2005), 684–698.

[60] Li, C.-G., Guo, J., and Zhang, H.-G. Local sparse representation based classification. In *2010 20th International Conference on Pattern Recognition* (Aug 2010), pp. 649–652.

[61] Li, Q., Li, T., Zhu, S., and Kambhamettu, C. Improving medical/biological data classification performance by wavelet preprocessing. In *2002 IEEE International Conference on Data Mining* (2002), pp. 657–660.

[62] Little, A. V., Maggioni, M., and Rosasco, L. Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. *Appl. Comput. Harmon. Anal* (2016), in press.

[63] Lowe, D. G. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision* (1999), vol. 2, IEEE Computer Society, pp. 1150–1157.

[64] Lustig, M., Donoho, D. L., Santos, J. M., and Pauly, J. M. Compressed sensing MRI. *IEEE Signal Processing Magazine 25*, 2 (March 2008), 72–82.

[65] Martinez, A., and Benavente, R. The AR face database. Tech. Rep. 24, Computer Vision Center, June 1998.

[66] Meng, H., Pears, N., and Bailey, C. Human action classification using SVM_2K classifier on motion features. In *Multimedia Content Representation, Classification and Security*, B. Gunsel, A. Jain, A. Tekalp, and B. Sankur, Eds., vol. 4105 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2006, pp. 458–465.

[67] Merkwirth, C., Parlitz, U., and Lauterborn, W. Fast nearest-neighbor searching for nonlinear signal processing. *Phys. Rev. E 62* (2000), 2089–2097.

[68] Merkwirth, C., Parlitz, U., Wedekind, I., Engster, D., and Lauterborn, W. TSTOOL homepage. `http://www.physik3.gwdg.de/tstool/index.html`, accessed 6.2.15. 2009.

[69] Niyogi, P., Smale, S., and Weinberger, S. Finding the homology of submanifolds with high confidence from random samples. *Discrete Comput. Geom. 39*, 1-3 (2008), 419–441.

[70] Ojala, T., Pietikainen, M., and Harwood, D. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In *Pattern Recognition, 1994. Conference A: Proceedings of the 12th IAPR International Conference on Computer Vision and Image Processing* (1994), vol. 1, pp. 582–585.

[71] Patel, R., Rathod, N., and Shah, A. Comparative analysis of face recognition approaches: A survey. *International Journal of Computer Applications 57*, 17 (Nov 2012), 50–69.

[72] Pennebaker, W. B., and Mitchell, J. L. *JPEG: Still Image Data Compression Standard*, 1st ed. Kluwer Academic Publishers, Norwell, MA, USA, 1992.

[73] Qiao, L., Chen, S., and Tan, X. Sparsity preserving projections with applications to face recognition. *Pattern Recognition 43*, 1 (2010), 331–341.

[74] Radhakrishna Rao, C. The utilization of multiple measurements in problems of biological classification. *J. Roy. Statist. Soc. Ser. B. 10* (1948), 159–193.

[75] Ragel, A., and Crémilleux, B. MVC—a preprocessing method to deal with missing values. *Knowledge-Based Systems 12*, 5-6 (1999), 285 – 291.

[76] Ray, A., Shukla, K., Aggarwal, L., Sharma, N., Pradhan, S., and Sharma, S. Segmentation and classification of medical images using texture-primitive features: Application of BAM-type artificial neural network. *Journal of Medical Physics 33*, 3 (2008), 119–126.

[77] Rosenfeld, M. In praise of the Gram matrix. In *The mathematics of Paul Erdős, II*, vol. 14 of *Algorithms Combin.* Springer, Berlin, 1997, pp. 318–323.

[78] Roweis, S. T., and Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science 290* (2000), 2323–2326.

[79] Saito, N. *Local Feature Extraction and Its Applications Using a Library of Bases*. PhD thesis, Department of Mathematics, Yale University, Dec 1994.

[80] Saito, N., and Coifman, R. R. Local discriminant bases and their applications. *J. Math. Imaging Vis. 5*, 4 (1995), 337–358. Invited paper.

[81] Seung, H. S., and Lee, D. D. The manifold ways of perception. *Science 290*, 5500 (2000), 2268–2269.

[82] Shannon, C. E. Communication in the presence of noise. *Proc. I.R.E. 37* (1949), 10–21.

[83] Sharon, Y., Wright, J., and Ma, Y. Minimum sum of distances estimator: Robustness and stability. In *2009 American Control Conference* (June 2009), pp. 524–530.

[84] Shawe-Taylor, J., and Cristianini, N. Estimating the moments of a random vector with applications. In *Proceedings of GRETSI 2003 Conference*, vol. I. 2003, pp. 47–52. Invited talk.

[85] Simard, P. Y., LeCun, Y. A., Denker, J. S., and Victorri, B. *Neural Networks: Tricks of the Trade: Second Edition*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, ch. Transformation Invariance in Pattern Recognition – Tangent Distance and Tangent Propagation, pp. 235–269.

[86] Simon, N., Friedman, J., Hastie, T., and Tibshirani, R. Regularization paths for Cox's proportional hazards model via coordinate descent. *Journal of Statistical Software 39*, 5 (2011), 1–13.

[87] Singer, A., and Wu, H.-T. Vector diffusion maps and the connection Laplacian. *Comm. Pure Appl. Math. 65*, 8 (2012), 1067–1144.

[88] Sprechmann, P., Ramírez, I., Sapiro, G., and Eldar, Y. C. C-HiLasso: a collaborative hierarchical sparse modeling framework. *IEEE Trans. Signal Process. 59*, 9 (2011), 4183–4198.

[89] Stewart, G. W., and Sun, J. G. *Matrix Perturbation Theory*. Computer Science and Scientific Computing. Academic Press, Inc., Boston, MA, 1990.

[90] Suo, Y., Dao, M., Tran, T., Mousavi, H., Srinivas, U., and Monga, V. Group structured dirty dictionary learning for classification. In *2014 IEEE International Conference on Image Processing (ICIP)* (Oct 2014), pp. 150–154.

[91] Timofte, R., and Gool, L. V. Adaptive and weighted collaborative representations for image classification. *Pattern Recognition Letters 43* (2014), 127 – 135.

[92] Tropp, J. A. On the conditioning of random subdictionaries. *Appl. Comput. Harmon. Anal. 25*, 1 (2008), 1–24.

[93] Tropp, J. A. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math. 12*, 4 (2012), 389–434.

[94] Tyagi, H., Vural, E., and Frossard, P. Tangent space estimation for smooth embeddings of Riemannian manifolds. *Inf. Inference 2*, 1 (2013), 69–114.

[95] van den Berg, E., and Friedlander, M. P. SPGL1: A solver for large-scale sparse reconstruction, June 2007. Version 1.9, April 2015 (accessed 12.4.2016).

[96] van den Berg, E., and Friedlander, M. P. Probing the Pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing 31*, 2 (2008), 890–912.

[97] Vershynin, R. Introduction to the non-asymptotic analysis of random matrices. In *Compressed Sensing*. Cambridge Univ. Press, Cambridge, 2012, pp. 210–268.

[98] Wang, Z., Yang, J., Nasrabadi, N., and Huang, T. A max-margin perspective on sparse representation-based classification. In *2013 IEEE International Conference on Computer Vision* (Dec 2013), pp. 1217–1224.

[99] Waqas, J., Yi, Z., and Zhang, L. Collaborative neighbor representation based classification using l_2-minimization approach. *Pattern Recognition Letters 34*, 2 (2013), 201 – 208.

[100] Weaver, C., and Saito, N. Improving sparse representation-based classification using local principal component analysis. *CoRR abs/1204.2358* (2016). Submitted for publication.

[101] Wei, C.-P., Chao, Y.-W., Yeh, Y.-R., and Wang, Y.-C. F. Locality-sensitive dictionary learning for sparse representation based classification. *Pattern Recognition 46*, 5 (2013), 1277–1287.

[102] Wright, J., and Ma, Y. Dense error correction via ℓ^1-minimization. *IEEE Trans. Inform. Theory 56*, 7 (2010), 3540–3560.

[103] Wright, J., Ma, Y., Mairal, J., Sapiro, G., Huang, T. S., and Yan, S. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE 98*, 6 (June 2010), 1031–1044.

[104] Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell. 31*, 2 (2009), 210–227.

[105] Xiaoyan, Z., Houjun, W., and Zhijian, D. Wireless sensor networks based on compressed sensing. In *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)* (July 2010), vol. 9, pp. 90–92.

[106] Xu, Y., Li, X., Yang, J., and Zhang, D. Integrate the original face image and its mirror image for face recognition. *Neurocomputing 131* (2014), 191–199.

[107] Yang, A. Y., Sastry, S. S., Ganesh, A., and Ma, Y. Fast ℓ1-minimization algorithms and an application in robust face recognition: A review. In *2010 17th IEEE International Conference on Image Processing* (Sept 2010), pp. 1849–1852.

[108] Yang, J., Wang, J., and Huang, T. Learning the sparse representation for classification. In *2011 IEEE International Conference on Multimedia and Expo (ICME)* (July 2011), pp. 1–6.

[109] Yang, J., Zhang, L., Xu, Y., and Yang, J.-Y. Beyond sparsity: The role of L1-optimizer in pattern classification. *Pattern Recognition 45*, 3 (2012), 1104–1118.

[110] Yang, J., Zhu, K., and Zhong, N. Local tangent distances for classification problems. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT)* (Dec 2012), vol. 1, pp. 396–401.

[111] Yin, J., Liu, Z., Jin, Z., and Yang, W. Kernel sparse representation based classification. *Neurocomputing 77*, 1 (2012), 120 – 128.

[112] Yuan, M., and Lin, Y. Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Ser. B Stat. Methodol. 68*, 1 (2006), 49–67.

[113] Zhang, H., Wang, F., Chen, Y., Zhang, W., Wang, K., and Liu, J. Sample pair based sparse representation classification for face recognition. *Expert Systems with Applications 45* (2016), 352 – 358.

[114] Zhang, L., Yang, M., and Feng, X. Sparse representation or collaborative representation: Which helps face recognition? In *Proceedings of the 2011 International Conference on Computer Vision* (Dec 2011), IEEE Computer Society, pp. 471–478.

[115] Zhang, L., Yang, M., Feng, X., Ma, Y., and Zhang, D. Collaborative representation based classification for face recognition. *CoRR abs/1204.2358* (2012).

[116] Zhang, L., Zhou, W.-D., Chang, P.-C., Liu, J., Yan, Z., Wang, T., and Li, F.-Z. Kernel sparse representation-based classifier. *IEEE Trans. Signal Process. 60*, 4 (2012), 1684–1695.

[117] Zhang, X., Pham, D.-S., Venkatesh, S., Liu, W., and Phung, D. Mixed-norm sparse representation for multiview face recognition. *Pattern Recognition 48*, 9 (2015), 2935 – 2946.

[118] Zhou, Y., Gao, J., and Barner, K. E. An enhanced sparse representation strategy for signal classification. In *Compressive Sensing*, vol. 8365. SPIE, 2012.

[119] Zhu, X., and Goldberg, A. *Introduction to Semi-Supervised Learning*. Morgan & Claypool, 2009.