ABSTRACT

Title of Dissertation: GRAPH-BASED DATA FUSION
WITH APPLICATIONS TO
MAGNETIC RESONANCE IMAGING

Jeremiah Emidih
Doctor of Philosophy, 2023

Dissertation Directed by: Professor Wojciech Czaja
Department of Mathematics

This thesis is concerned with development and applications of data fusion methods in the context of Laplacian eigenmaps. Multimodal data can be challenging to work with using classical statistical and signal processing techniques. Graphs provide a reference frame for the study of otherwise structure-less data. We combine spectral methods on graphs and geometric data analysis in order to create a novel data fusion model. We also provide examples of applications of this model to bioinformatics, color transformation and superresolution, and magnetic resonance imaging.

GRAPH-BASED DATA FUSION
WITH APPLICATIONS TO MAGNETIC RESONANCE IMAGING

by

Jeremiah Emidih

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2023

Advisory Committee:
Professor Wojciech Czaja, Chair/Advisor
Dr. Richard G. Spencer
Professor John Benedetto
Professor Vince Lyzinski
Professor Rob Patro, Dean's Representative

Dedication

In memory of my father.

I wish you could have been here for the ending, I know you would be proud.

Acknowledgments

First, I thank my advisor, Professor Wojciech Czaja, a patient, knowledgeable, and trusted mentor through my struggles with the challenges of academia. Thanks to Dr. Richard Spencer, my de facto co-advisor, for his guidance in research and his ability to drag a horse to water on multiple occasions. I thank the rest of my committee, Professors John Benedetto, Vince Lyzinski, and Rob Patro, for spending time reviewing this thesis. In particular, I thank Professor Benedetto for introducing me to the field of harmonic analysis and to the framework of *applicable* mathematics.

Thanks to my fellow students, including Kasun Fernando, Pratima Hebbar, Micah Goldblum, David Pincus, J.T. Rustad, Steven Reich, and Asia Wyatt, for helpful collaboration and valued friendship. I also recognize alumni whose work precedes my own: Alexander Cloninger, Timothy Doster, Ariel Hafftka, Weilin Li, James Murphy, and Vinodh Rajapakse. A special mention goes to Nadine Jansen, Candice Price, Shelly Scott, Caprice Stanley, and Professor Kasso Okoudjou for expanding my scope of who a mathematician can be.

Finally, I am indebted to my friends and loved ones for listening to my complaints for the better part of a decade. To Teta, I truly couldn't have done this without you in my ear and heart. To my brother, thank you for always meeting my mercurial curiosity with matched enthusiasm. To my mother, I'm happy that you were correct about how this would turn out, a direct result of your support and prayer. Also, thank you to the music of Stevie Wonder.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1: Introduction

The success of machine learning techniques over the last two decades has brought about a sea change in the analysis and synthesis of data. As a part of this dramatic increase in attention and effort in data science and related fields, integration of data from different sources and across modalities has become a prominent and interdisciplinary research topic, largely powered by advances in *neural networks*. However, the positive results of these methods rely heavily on the type and volume of data available. While natural images are abundant in large volumes and satisfy many reasonable mathematical assumptions, making tools like convolutional neural networks viable tools for analysis, specialized data such as medical images appropriate for a particular application can be much more expensive to produce or otherwise difficult to obtain. These difficulties amount to inadequate training data for the essential parameter optimization which powers modern deep learning algorithms. For these reasons, we are interested in investigating data-dependant machine learning methods which will prove useful across a variety of domains.

In response to the profusion of multimodal data, both in modern life and in scientific study, we utilize a range of machine learning methods in order to develop our primary contribution, a novel graph-based data fusion model. Our work demonstrates the feasibility of this model to the problem of integrating multimodal image data. As part of this model, we contribute to the field of geometric data analysis with the introduction of a clustering method designed for Laplacian

eigenmaps, a foundational technique in the burgeoning field of data analysis on graphs. Our model builds upon literature in outlier detection, shape analysis, and graph matching to provide a framework for mapping between feature spaces of multimodal data.

We now provide an overview of the work presented in this thesis.

First, we discuss magnetic resonance imaging (MRI) in Chapter 2, starting with an explanation of how such images are acquired. We present the motivation for our main application, a superresolution technique for multiple MRI modalities. This application has garnered a great deal of attention and research recently, and we discuss some relevant literature and common data sets.

Next, we provide the mathematical foundations for data analysis with kernel methods in Chapter 3. We progress through methods of mathematical methods in machine learning, and give an exposition of Laplacian eigenmaps from multiple perspectives. Several related algorithms which feature heavily in our research are also explored. The chapter concludes with a description of how images can be analyzed with such tools.

In Chapter 4, we present our main contribution, the development of a novel data fusion model, based around Laplacian eigenmaps. We begin with a curvature-based clustering method and follow that with mappings between data embedding spaces based on graph matching ideas. We present the general framework of the model, and prepare the stage for interesting applications of the model.

Then, in Chapter 5, we present an early application, essential in developing our current data fusion paradigm. This chapter documents the search through high-dimensional genomic data for predictors of drug response to certain cancer cells using Laplacian eigenmaps and data fusion as an outlier detection method. This also establishes the usefulness of the data fusion method in

supervised learning problems.

Chapter 6 introduces the application of our data fusion model to image analysis. We describe experiments using the model to recover color pixel information from grayscale images in the CIFAR-10 data set. This is the first step in refining our novel multimodal superresolution (MMSR) algorithm, powered by our data fusion framework.

Finally, in Chapter 7, we adapt and apply this multimodal superresolution technique in the context of MRI. Inspired by recent image analysis methods and the physical systems governing magnetic resonance, we develop a method for superresolution of MRI, aided with multimodal data. Our method is tested on publically available, clinical MRI data and compared to standard methods.

Chapter 2: Motivation and Imaging Background

2.1 Description of MRI

The primary application of my research is to magnetic resonance imaging (MRI), with the related experiments documented in Chapter 7. In this section, we give a brief overview of MRI at a level sufficient to provide necessary background to those analyses. Much of the material in this section is based closely on the widely used textbook [63] by Hashemi, with further details to be found in that text, with more advanced material also available in a graduate-level textbook by Haacke et al. [20]. First, we give a description of the process by which MR images are obtained, with particular attention to the generation of so-called *T1-weighted (T1w)* and *T2-weighted (T2w)* images which are the focus of the applied work in this thesis, and then describe some of the features of T1w and T2w brain images. We first note that MRI maps the protons of water molecules, and is particularly sensitive to differences in mobility of water molecules in different tissues and regions of tissue. In fact, along with density, these processes form the main mechanisms for generating contrast in an MR image.

An MRI machine contains both a large magnet, whose uniform magnetic field is referred to as $\mathbf{B_0}$, and a radio frequency (RF) coil, capable of sending and receiving signals within the radio frequency range. The central concept of MRI is to apply a sequence of RF electromagnetic pulses to an object, apply magnetic field gradient pulses at appropriate times, and record the signal that

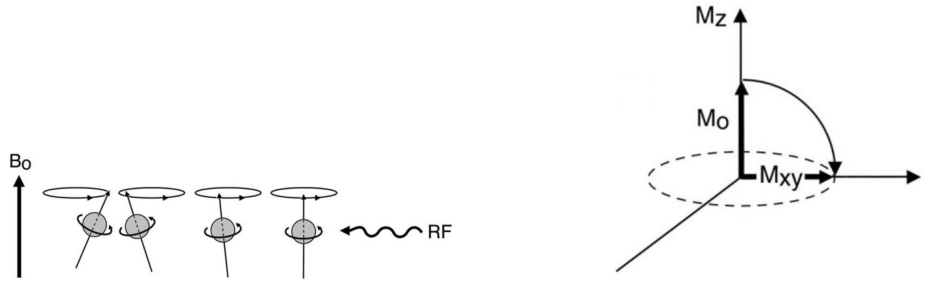
Figure 2.1: *Image of a Siemens Magnetom Prisma 3T MRI Scanner at the Maryland Neuroimaging Center, University of Maryland*

Image Source: Maryland Neuroimaging Center, UMD

arises in the form of the electromagnetic wave radiated by that object. The RF pulses act on the proton within each of the two hydrogen atoms in a water molecule by inducing transitions between what may be called a "spin up" and "spin down" state with respect to the main magnetic field of the MR system. This field ensures that there is an overpopulation of one of these spin orientations. The lower energy orientation aligns parallel to the $\mathbf{B_0}$ field, leading to a net magnetic dipole moment that is detected in the acquired signal.

The first step in obtaining an MR signal is to place some object into the magnetic field $\mathbf{B_0}$ and apply an RF pulse from the RF coil. This pulse is applied in such a way as to cause the net magnetization to point in a new direction, called the transverse direction, perpendicular to the direction of $\mathbf{B_0}$, which is known as the longitudinal direction. At the same time, the RF pulse preserves the small net overabundance of spins parallel to, rather than antiparallel to, the $\mathbf{B_0}$ field.

Figure 2.2: *A radio frequency (RF) pulse acts on protons (a) which have been exposed to magnetic field* $\mathbf{B_0}$*. As a result of the action of the RF pulse (b), the net magnetization vector lies on the transverse plane.*



(a) *RF Pulse* **(b)** *Magnetization vector*

Image Source: "MRI Basics" by Hashemi [63]

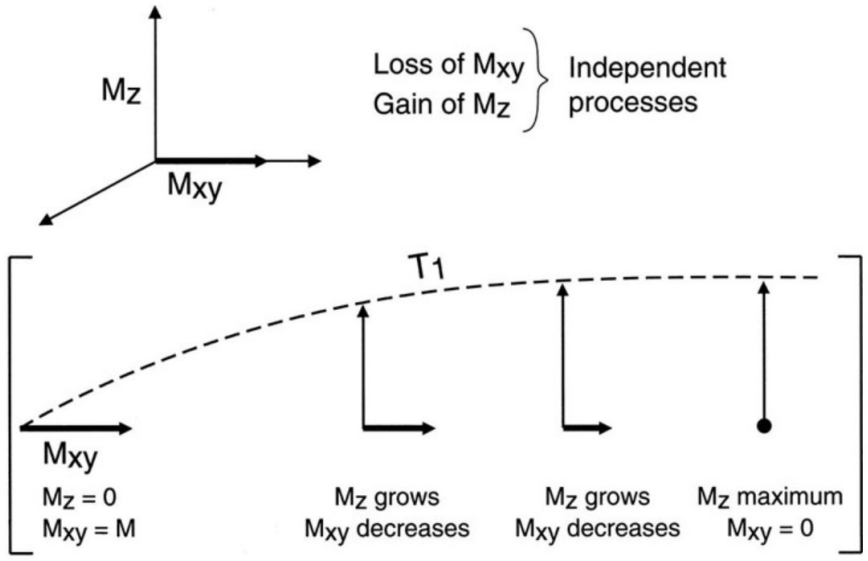
After the brief RF pulse, on the order of a few milliseconds, the magnetization in the transverse direction decays in the transverse plane over a timescale of tens of milliseconds; this is called *T2, or transverse, relaxation*. Simultaneously, but on the order of hundreds of milliseconds,

the magnetization in the longitudinal direction, that is, along the main magnetic field $\mathbf{B_0}$, grows in a distinct process known as *T1, or longitudinal, relaxation*. These decay and growth time constants are referred to simply as T2 and T1.

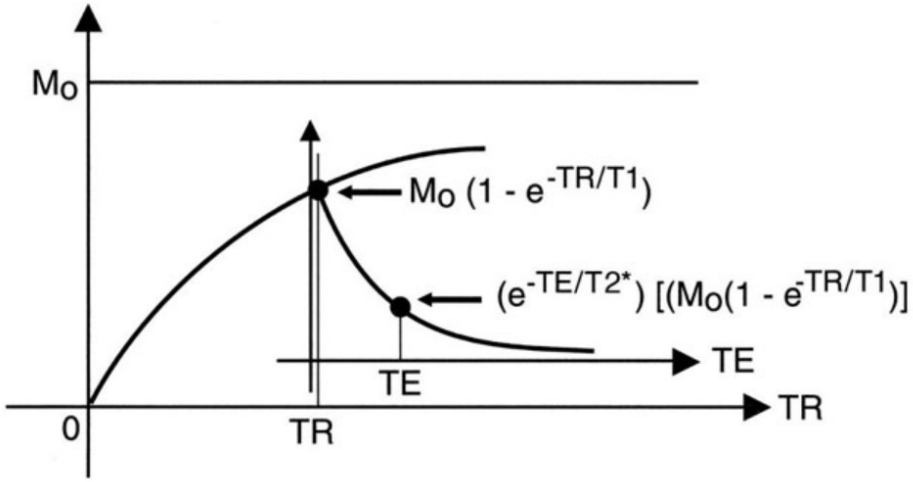
As the transverse decay and longitudinal growth processes occur, the object being imaged emits a weak electromagnetic signal generated by protons precessing in the main magnetic field. This process is what creates the relationship between $\mathbf{B_0}$ and the MRI detection frequency. The amount of signal recovered by the RF coil at the time of signal acquisition, known as the echo time (TE), then depends on the current degree of proton magnetic moment transverse relaxation and longitudinal growth, and hence on the T1 and T2 values for that particular tissue or region of tissue. Controlling both the sequence of acquisition times used, that is, TE values, and the time before repeated RF pulses, known as the repetition time (TR), allows the user to generate signals so that differences in quantities such as density, T1, or T2 between tissues or regions of tissue are the main determinants of signal strength. A density-weighted, *T1-weighted (T1w)* or *T2-weighted (T2w)* image is then the image created from a density, T1 or T2 weighted signal, respectively. The image comes as a result of a certain rescaling of the discrete Fourier transform of the recorded signal as the magnetization precesses in the transverse plane. In this thesis, we use the terms "T1" or "T2" in reference to physical properties of a particular object. On the other hand, we use "T1w" or "T2w" to describe the different image modalities acquired when the MRI signal is weighted to represent more of an object's T1 or T2, respectively.

The rescaling noted above results from application of a known magnetic field gradient across the object, leading to a known frequency variation in the acquired object, leading to a known position within the object from which the signal is detected. In brief, this permits spatial assignment of a frequency component, readily detected through a Fourier transform of the signal,

Figure 2.3: *After the RF pulse is turned off (a), magnetization decays in the transverse direction and grows in the longitudinal direction. In (b), recovery and decay curves are plotted on the same graph. The echo time (TE) and repetition time (TR) are on different scales.*



(a) *Relaxation after RF pulse*



(b) *Growth and decay curves*

Image Source: "MRI Basics" by Hashemi [63]

to a spatial position. This allows for the formation of MR signal strength maps, or MR images. The strength of this signal is dependent on density, T1, and T2, as described above, yielding images with the desired contrast. A similar localization process results in restriction of acquired data to a particular cross-sectional slice, as depicted in Figure 2.4.

Figure 2.4: *A diagram depicting MRI slice acquisition. The magnetic field has a spatial gradient and the field strength is linearly proportional to the frequency of the RF pulse. This allows for spatial localization.*

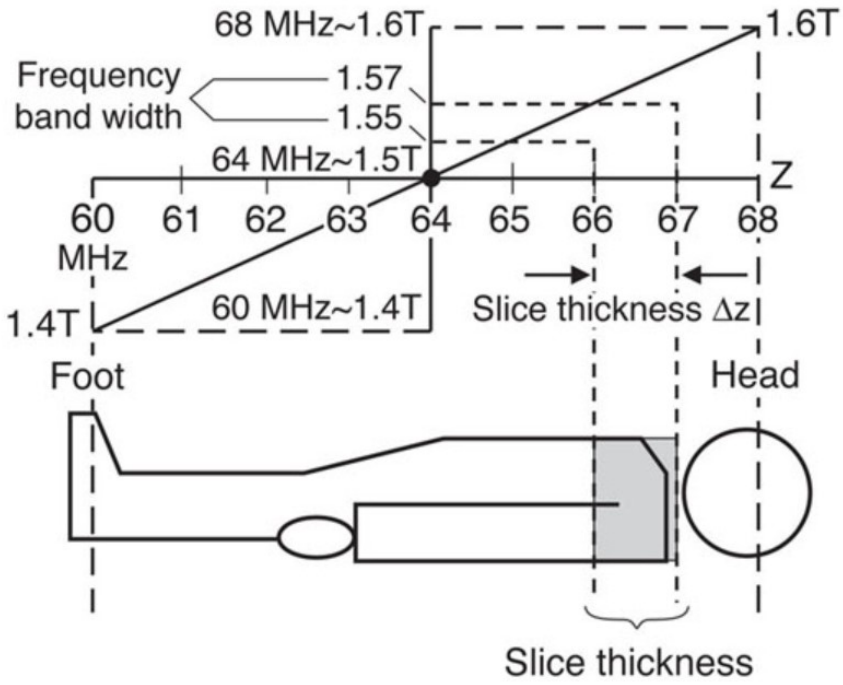


Image Source: "MRI Basics" by Hashemi [63]

The T1 or T2 of an object is an intrinsic property of the material and as such, the contrast in a T1w or T2w image reflects the type of tissue present at a location. T2 measures how fast hydrogen protons come out of phase, or dephase, after the RF pulse is turned off. Fluids like water and cerebrospinal fluid (CSF) have molecules which are highly mobile so their dipole-dipole interactions are averaged out, leading to a longer time constant for spin dephasing and

thus signal loss. Because of this, these fluids appear with high intensity in T2w images. On the other hand, white matter in the brain have molecules which are less mobile and hence interact more strongly with each other, generating greater average dipole-dipole interactions and leading to more rapid transverse decay as compared to CSF. Similarly, gray matter contains water protons which are noticeably more mobile than those in white matter. Overall, on a T2w image, CSF appears much brighter than white or gray matter, while gray matter appears slightly lighter than white matter. This results in lower intensity pixels of white matter in T2w images. The T1 of an object determines how fast its protons realign with the field $\mathbf{B_0}$, which is governed by how easily the protons spread energy between themselves and surrounding tissue. These processes are also slower for mobile tissues, which means that there will be less signal recovery between RF pulses for a given value of TR. This means that we see CSF as dark in T1w images as compared to white and gray matter, while gray matter appears somewhat darker than white matter. Figure 2.5 demonstrates this image contrast between tissues in T1w and T2w images.

2.2 Motivation

Acquiring high resolution MRI comes at the cost of either decreased *signal-to-noise ratio* (SNR), increased signal acquisition time, or perhaps both [96]. This is because acquisition of a finer pixel grid requires more pulses, each with higher relaxation time occurring before application. Consequently, we are motivated by the fact that higher resolution MRI are slower to acquire for a given SNR, critical in a clinical setting. Clinicians are thus directed toward choosing low resolution over poorer SNR or increase in scan time. T1w scans can often take longer to acquire than T2w scans depending on their pulse parameters, or vice versa [63]. An ideal strategy would

then be to acquire a low resolution MRI and, under certain conditions, increase the resolution after image acquisition.

Superresolution (SR), in brief, refers to the mathematical problem of inducing such an increase in resolution in an image, generating a higher resolution image from a lower resolution image. Here, we make a distinction between *single image superresolution* (SISR), in which a high resolution image is created from a low resolution version without additional information, and *multimodal superresolution* (MMSR), where the SR problem is aided by the inclusion of a high resolution image of a different, related modality. Over the last decade, there has been much research interest in either some variant of an SISR problem in MRI, the MMSR problem involving transfer of information between modalities, or a combination of the two [23, 44, 83, 85, 86, 96, 108, 124, 125]. Lyu et al. [85] constructed a *convolutional neural network* (CNN) to create high resolution T2w images from either T1w images or *proton-density* (PD) weighted images. Sharma et al. [108] utilize a *generative adversarial network* (GAN) to synthesize multimodal MRI in 4 modalities using combined images from a subset of those image types. Dai et al. [44] continue in that direction, synthesizing multimodal MRI using only a single modality via a GAN. While much of the recent literature is based on recent *deep learning* constructions such as CNNs or GANs, those tools are limited by the amount of training data required for suitable results. This is less of an issue with nonmedical natural image processing tasks, but with MRI, large and varied data sets are not as readily available. Our work takes this into consideration, as we use *kernel-based methods* in our approach to an MMSR problem. Other work in this area include research by Zheng et al. [124]. They develop a method for solving an MMSR problem using a model which assumes that image contrast of one modality can inform and enhance contrast of another. While our method also extrapolates information from pixel intensity differences, we use a combined

spectral-spatial technique to measure differences outside of spatial neighborhoods. Lu et al. [83] use a similar kernel method to ours, using a *wavelet*-based image fusion technique along with *local linear embedding*. The local linear embedding technique shares many attributes with our preferred *Laplacian eigenmaps* technique, but our method uses a simpler joint image construction when compared to their wavelet-based image fusion. Zhu et al. [125] combine a deep learning approach with a low-dimensional manifold model in order to transfer between MRI modalities. In comparison to our problem, Zhu et al. use the raw data from the MRI machine itself, whereas our method only requires access to acquired images.

In the applications described in this thesis, we rely heavily on the fact that T1w and T2w images portray the same structures in different manners. We will develop data-dependent, graph-based techniques for solving the multimodal superresolution problem, with a focus on reconstruction of high resolution T1w images from a combination high resolution T2w and low resolution T1w images. To our knowledge, such techniques have not been employed in this MRI MMSR problem before, and so our work provides a novel contribution toward the problem.

2.3 Description of MR Data Sets

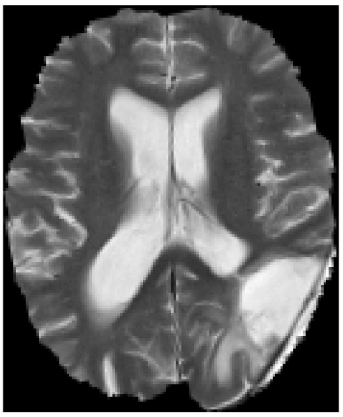
The MRI featured in this thesis are mostly sourced from one of two data sets, one consisting of clinical, anonymized brain scans and another consisting of synthetic images which are meant to model brain anatomy.

2.3.1 BraTS Data Set

The Brain Tumor Segmentation (BraTS) Challenge is an annual competition, lead by Spyridon Bakas of the University of Pennsylvania Perelman School of Medicine, which focuses on segmentation of brain tumors and tumor subregions [7, 8, 88]. Gliomas prove especially challenging for detection by automatic methods due to their tissue heterogeneity and hence, high variance in image profiles in different MR modalities [9]. The BraTS data set from the 2017 competition features 484 training volumes along with training labels for each voxel delineating tumor and surrounding edema [5, 6]. Each volume is made of 240 by 240 by 155 voxels given in both T1w and T2w scans. The different modalities are each registered to the same anatomical model per image, so voxels at the same coordinates in the same brain for different modalities capture the same physical location.

Figure 2.5: *Example images from BraTS data set. Note that the CSF in the central ventricles appears bright (b) in T2w, but is dark (a) in T2w.*

(a) *T1w Example from BraTS, Axial View*



(b) *T2w Example from BraTS, Axial View*

2.3.2 BrainWeb Database

The BrainWeb Simulated Brain Database, known as BrainWeb, is an online database of synthesized MRI volumes from an MRI simulator [33, 71, 72]. BrainWeb is based on two anatomical models, one normal and one containing multiple sclerosis (MS) lesions. These images are created by combining anatomical models of 20 different brain tissues, such as white matter or fat, depicted as T1w or T2w images. Of course, these models of the brain cannot fully encompass the complexity of actual brain scans, healthy or otherwise. In our experiments, described in Chapter 7, we will emphasize analysis on the BraTS data set for this reason. Nevertheless, BrainWeb has been used for testing and initial validation of various medical imaging tasks, including early iterations of the BraTS competition [9, 87]. We choose to work with only two image volumes in this database, one from each of the two available models, as other available volumes vary mostly in amounts of noise added or modeled MR acquisition techniques. The volumes are made up of 217 by 181 by 181 voxels, and they are spatially registered as they are generated using the same basic anatomical template.

Chapter 3: Fundamentals

3.1 Outline

In this chapter, we will describe the mathematical foundations supporting the work of this thesis. We begin with a thorough explanation of Laplacian eigenmaps, linking the kernel-based construction with that of spectral graph theory. We then detail how to use Laplacian eigenmaps and related methods for data analysis and integration, with special focus on natural images.

3.2 Kernel PCA to Laplacian Eigenmaps

The bulk of the material in this section is sourced from the primary source literature on the discussed techniques [10, 67, 82, 94, 105, 116], the textbook on Dimensionality Reduction by Lee and Verleysen [74], and the textbook on Statistical Learning by Hastie, Tibshirani, and Friedman [64]. The paper developing explaining the link between these kernel methods by Bengio, Vincent, and Paiement [18] was a particularly helpful source as well.

3.2.1 Principal Component Analysis

Principal component analysis (PCA) is a classic statistical learning technique in which a data set can be transformed into a space whose basis vectors are oriented in the directions of

maximum variance of that set. Originally developed by Pearson in 1901 [94], then further refined separately by Karhunen [67] and Loève [82], this method has proven useful for exploratory data analysis. PCA is a relatively computationally efficient method of unsupervised data exploration and can be computed using either an eigenvalue decomposition (EVD) or a singular value decomposition (SVD) on certain matrices which represent the data set.

We begin with a brief summary of PCA. Let $X \in \mathbb{R}^{n \times D}$ be a data matrix consisting of n observations, considered as rows of X, in some D-dimensional space. Further, assume that the data is mean-centered, so that each of the D column feature vectors has mean 0. The driving assumption for PCA is that there exists some set of d orthonormal vectors $\{v_1, \ldots, v_d\}, v_i \in \mathbb{R}^D$ for all $1 \leq i \leq d$, which serves as a basis of a *latent space* in which the data set is more efficiently represented, in terms of number of components. Denote the matrix $V = [v_1, \ldots, v_d]^\top \in \mathbb{R}^{d \times D}$ as the matrix of these basis vectors as rows. We assume then that X is the result of a linear transformation of a set of latent data vectors in the matrix $Z = [z_1, \ldots, z_n]^\top \in \mathbb{R}^{n \times d}$ and that this linear transformation is expressible as $X = ZV$. Assume further that each of the latent data vectors is multivariate Gaussian distributed with zero mean.

As V is assumed to have orthonormal rows, $VV^\top = I_d$. In the idealized situation, we would have $X = ZV \implies XV^\top = ZVV^\top = Z$. The basis vectors which form V are then found in an optimization problem to minimize the least squares reconstruction error of X from Z. We can rewrite the reconstruction error $\|X - XV^\top V\|_F^2$ as follows:

$$\|X - XV^\top V\|_F^2 = \|X^\top - V^\top V X^\top\|_F^2 = tr((X^\top - V^\top V X^\top)^\top (X^\top - V^\top V X^\top)) =$$

$$tr(X^\top X - 2XV^\top V X^\top + XV^\top V V^\top V X^\top) =$$

$$= tr(X^\top X) - tr(XV^\top VX^\top) = \sum_{i=1}^{n} (x_i^\top x_i - x_i V^\top V x_i^\top).$$

Hence, the matrix V which minimizes the reconstruction error is the one which maximizes the sum of all terms $x_i V^\top V x_i^\top$ over each data point $x_i \in \mathbb{R}^D$. To solve the resultant optimization problem, let $X = U\Sigma W^\top$ be the singular value decomposition (SVD) of X, where the singular values in Σ are written in descending order. Then $V = I_{d\times D} W^\top$ minimizes the reconstruction error, with $I_{d\times D} \in \mathbb{R}^{d\times D}$ as the matrix with 1 down its main diagonal and 0 everywhere else, since

$$\|X - XV^\top V\|_F^2 = \|U\Sigma W^\top - U\Sigma W^\top W I_{D\times d} I_{d\times D} W^\top\|_F^2 = \|U\Sigma W^\top - U\Sigma W^\top\|_F^2 = 0.$$

We then refer to the latent basis vectors, the rows of $V = I_{d\times D} W^\top$ as the *principal components* of the data set X. The projection of X onto the space of principal components is given by $\widehat{X} = XV^\top = XW I_{D\times d}$. In other words, principal components transformation is a projection on to the d singular vectors corresponding to the d-largest singular values of X.

There is an alternative formulation of PCA which leverages the other foundational property which makes the transformation useful. We again assume that the data matrix $X \in \mathbb{R}^{n\times D}$ is the result of a linear transformation of latent data vectors $Z = [z_1, \ldots, z_n]^\top \in \mathbb{R}^{n\times d}$. This time, we demonstrate that the transformation under V, the matrix composed of orthonormal latent basis vectors as rows, $X = ZV$ is such that the maximum variance is preserved and the rows of Z are uncorrelated.

The sample covariance matrix C_{XX} of X is given by $\frac{1}{n} X^\top X$. As a positive definite matrix, C_{XX} admits an eigenvalue decomposition (EVD) with orthonormal eigenvectors, so we

let $C_{XX} = \frac{1}{n}X^\top X = \frac{1}{n}P\Lambda P^\top$ with $P \in \mathbb{R}^{D\times D}$ orthogonal and $\Lambda \in \mathbb{R}^{D\times D}$ diagonal with eigenvalues written in decreasing order. At the same time, the SVD of X gives that $C_{XX} = \frac{1}{n}W\Sigma^\top U^\top U\Sigma W^\top = W\Sigma^2 W^\top$. Both Λ and Σ^2 are diagonal and both P and W are orthogonal, so by the uniqueness of the EVD, we have $P = W$. Under the assumption $X = ZV$, the covariance matrix of Z is $C_{ZZ} = \frac{1}{n}Z^\top Z = \frac{1}{n}(XV^\top)^\top(XV^\top) = \frac{1}{n}VC_{XX}V^\top = (VW)\Sigma^2(VW)^\top$. Because the rows of Z are uncorrelated, it must be that rows of V are colinear with the first d-columns of W. We then choose $V = I_{d\times D}W^\top$ as the matrix of latent basis variables. Notice that these are the same as the principal components defined earlier. This is in part because uniqueness of the EVD means $\Sigma^2 = \Lambda$, which then implies that $tr(C_{XX}) = tr(C_{ZZ})$, so we retain the maximum sample variance property.

3.2.2 Multidimensional Scaling

This gives us two characterizations of principal components: as the orthogonal basis vectors which minimize reconstruction error under an orthogonal projection, derived as the solution to an optimization problem, and also as the uncorrelated orthogonal basis vectors which preserve the maximum variance. In particular, the second characterization can be reinterpreted by thinking of the rows of X as location vectors in $\mathbb{R}^D$, centered at the origin. This is the *multidimensional scaling* (MDS) perspective, first noted by Torgerson [116]. Under this framework, the matrix of study is no longer the covariance matrix, proportional to $X^\top X$, but a *Gram* matrix $XX^\top$ composed of all the pairwise inner products of the rows of X. In fact, one can start with the Gram matrix instead of X itself. Let V be defined as before, then

$$Z = XV^\top = XWI_{D\times d} = U\Sigma I_{D\times d}.$$

U can be realized as the left eigenvectors of the Gram matrix by the definition of the SVD of X, and if Λ is the square matrix of eigenvalues of $XX^\top$, then $\Sigma I_{D\times d} = \Lambda^{\frac{1}{2}} I_{n\times d}$ since the singular values of X are the eigenvalues of $XX^\top$. Thus, we can think of MDS as an alternate formulation of PCA, computed as an eigenvalue problem involving the Gram matrix.

3.2.3 Kernel PCA

For their relative simplicity as linear transformations, PCA and MDS are unable to separate data across nonlinear classifiers. Scholkopf [105] devised kernel PCA (kPCA) as a generalization to deal with this and adapt the ideas of PCA to allow for nonlinearities. Williams [122] later provided a more robust connection between kPCA and MDS.

The premise begins similarly to MDS: starting with a data matrix $X \in \mathbb{R}^{n\times D}$, find latent data vectors $Z = [z_1, \ldots, z_n]^\top \in \mathbb{R}^{n\times d}$. Prior to the MDS calculations however, we first embed X into a larger, potentially infinite dimensional, feature space $f(X) = Y$ via some embedding function f. MDS at this stage would then calculate eigenvectors of the Gram matrix, composed of the mutual inner products of the points in this feature space, in the hopes that what would have been a nonlinear classification in $\mathbb{R}^D$ can be tackled in this feature space. Calculation of the inner products of Y is generally unfeasible, and in fact, the actual function f is usually unknown. Instead, we require that the inner product in this feature space is expressible as a *kernel function* $\kappa : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, so that $\langle f(x_i), f(x_j)\rangle_{f(X)} = \kappa(x_i, x_j)$ for all $1 \le i, j \le n$.

We now briefly describe the conditions under which such a requirement is possible. κ must be the continuous kernel of a positive definite integral operator on $L^2(\Omega)$ functions $(\mathcal{K}f)(x) = \int_\Omega \kappa(x,t)f(t)dt$, where $\Omega \subseteq \mathbb{R}^D$ is a compact set. Then, by the spectral theorem, $\mathcal{K}$ admits

orthonormal eigenfunctions $\{e_i\}_{i=1}^{\infty}$, with nonnegative eigenvalues $\{\lambda_i\}_{i=1}^{\infty}$, which form an orthonormal basis of $L^2(\Omega)$. Mercer's theorem then states that $\kappa(x,t) = \sum_{i=1}^{\infty} \lambda_i e_i(x) e_i(t)$. Thus in the space generated by $\{e_i\}_{i=1}^{\infty}$, the inner product of two functions coincides with evaluation of κ.

In practice, κ is chosen usually as either some polynomial of the Euclidean inner product $\kappa(x_i, x_j) = p(\langle x_i, x_j \rangle)$ or as a radial basis function $\kappa(x_i, x_j) = f(\|x_i - x_j\|^2)$. If κ is the Euclidean inner product itself, then kernel PCA recovers regular PCA. From here, one can proceed with the MDS calculation, using a matrix of kernel values $K \in \mathbb{R}^{n \times n}$ instead of the Gram matrix of X. This means that the latent basis vectors Z are given by $Z = U \Lambda I_{n \times d}$, where $Z = U \Lambda U^{\top}$ is the eigendecomposition of the kernel matrix K.

3.2.4 Laplacian Eigenmaps as a Kernel Method

We will now investigate a formulation of kernel PCA, created by Belkin and Niyogi [10], for a particular family of kernel functions, establishing the primary mathematical object of study in this chapter.

First, we want to consider the situation in which data matrix $X \in \mathbb{R}^{n \times D}$ has a useful sense of *local connectivity*. That is, suppose there is some kernel function $w : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_{\geq 0}$ which measures the connection between rows of X. We would prefer w to be some radial basis function, but in general w is subject to the same constraints of a kernel function as mentioned in Section 3.2.3, so well will first define a few useful matrices. Create a matrix $W \in \mathbb{R}^{n \times n}$ with $W_{ij} = w(x_i, x_j)$ and let $D \in \mathbb{R}^{n \times n}$ be the diagonal matrix of row sums of W, i.e. $D_{ii} = \sum_{j=1}^{n} W_{ij}$. Define L as $L = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$. We will soon show that L is a suitable kernel matrix with

some interesting properties.

Next, we seek out a set of latent basis vectors again, this time chosen to respect local connectivity as given by matrix W. Let $Y = [y_1, \ldots, y_d] \in \mathbb{R}^{n \times d}$ be those basis vectors, which can also be thought of as maps for rows in X into some new embedding space. Then we can solve for Y via this optimization problem [10]:

$$Y = \underset{Y^\top Y = I_{d \times d}}{\operatorname{argmin}} \frac{1}{2} \sum_{p=1}^{d} \sum_{i,j=1}^{n} \left(\frac{y_p(i)}{\sqrt{D_{ii}}} - \frac{y_p(j)}{\sqrt{D_{jj}}} \right)^2 W_{ij} = \underset{Y^\top Y = I_{d \times d}}{\operatorname{argmin}} E_{LE}.$$

E_{LE} measures the sum of normalized coordinates, weighted by strength of local connection.

To solve this, we first rewrite each p-term E_{LE} to incorporate matrix L:

$$\begin{aligned}
\frac{1}{2} \sum_{i,j=1}^{n} \left(\frac{y_p(i)}{\sqrt{D_{ii}}} - \frac{y_p(j)}{\sqrt{D_{jj}}} \right)^2 W_{ij} &= \frac{1}{2} \sum_{i,j} \left(\frac{y_p^2(i)}{D_{ii}} W_{ij} + \frac{y_p^2(j)}{D_{jj}} W_{ij} - 2 \frac{y_p(i) y_p(j)}{\sqrt{D_{ii} D_{jj}}} W_{ij} \right) = \\
&= \frac{1}{2} \sum_{i} \frac{y_p^2(i)}{D_{ii}} D_{ii} + \frac{1}{2} \sum_{j} \frac{y_p^2(j)}{D_{jj}} D_{jj} - \sum_{i,j} \frac{y_p(i) y_p(j)}{\sqrt{D_{ii} D_{jj}}} W_{ij} = \\
&= 2 \left(\frac{1}{2} \sum_{i} y_p^2(i) \right) - \sum_{i,j} \frac{y_p(i) y_p(j)}{\sqrt{D_{ii} D_{jj}}} W_{ij} = \\
&= \sum_{i} \left(y_p^2(i) - \sum_{j} \frac{y_p(i) y_p(j)}{\sqrt{D_{ii} D_{jj}}} W_{ij} \right) \\
&= \sum_{i} y_p(i) \left(y_p(i) - \sum_{j} \frac{y_p(j)}{\sqrt{D_{ii} D_{jj}}} W_{ij} \right) \\
&= \langle y_p, (I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}) y_p \rangle \\
&= \langle y_p, L y_p \rangle.
\end{aligned}$$

As an aside, we now have that L is a positive semi-definite matrix, since each p-term of E_{LE} is nonnegative, as well as symmetric due to the symmetry of W. Hence, if the columns of

Y are mutually orthogonal with norm 1, then E_{LE} is the sum of *Rayleigh quotients* of L:

$$E_{LE} = \sum_{p=1}^{d} \frac{y_p^\top L y_p}{y_p^\top y_p} = \sum_{p=1}^{d} y_p^\top L y_p = Tr(Y^\top L Y).$$

Note that, by definition of matrix D, $W\mathbb{1} = D\mathbb{1}$, where $\mathbb{1} \in \mathbb{R}^n$ is the column vector of all 1s, as both have equivalent row sums. As a result,

$$\begin{aligned} L(D^{\frac{1}{2}}\mathbb{1}) &= D^{\frac{1}{2}}\mathbb{1} - \left(D^{-\frac{1}{2}} W D^{-\frac{1}{2}}\right)\left(D^{\frac{1}{2}}\mathbb{1}\right) = D^{\frac{1}{2}}\mathbb{1} - D^{-\frac{1}{2}}\left(W\mathbb{1}\right) \\ &= D^{\frac{1}{2}}\mathbb{1} - D^{-\frac{1}{2}}\left(D\mathbb{1}\right) = D^{\frac{1}{2}}\mathbb{1} - D^{\frac{1}{2}}\mathbb{1} = 0. \end{aligned}$$

Thus, $D^{\frac{1}{2}}\mathbb{1}$ is an eigenvector of L with eigenvalue 0. This is then the global minimizer of $y_p^\top L y_p$, but is usually not included as one of the basis vectors because it provides no new information relative to calculation of the kernel function itself. If we then add a further assumption that each y_p be orthogonal to the nullspace of L, then $y_p^\top L y_p$ is minimized by ϕ_1, the eigenvector of L with the smallest nonzero eigenvalue λ_1. The minimizer of $y_p^\top L y_p$ where y_p is orthogonal to both the nullspace of L and ϕ_1 is ϕ_2, the eigenvector of the second smallest nonzero eigenvalue λ_2. We can continue in this way to find Y as the matrix of eigenvectors of the d-smallest nonzero eigenvalues of L. These vectors are known as *Laplacian eigenmaps*. We will see in the next section that these have a connection to graph theory, as well as roots in Riemannian geometry.

If we take L to be a kernel matrix, then Laplacian eigenmaps can be realized as kernel principal components without scaling by eigenvalues. Since Laplacian eigenmaps uses the smallest eigenvalues, increasing up from 0, rather than the largest eigenvalues of some kernel matrix, it is disadvantageous to use the form $Y = U\Lambda I_{n\times D}$. The next section details another perspective on these maps, opening the pathway for data analysis which puts geometry at the forefront.

3.3 Spectral Graph Theory

This section primarily is sourced by the seminal book of the same name by Chung [25] and the original paper on Laplacian eigenmaps by Belkin and Niyogi [10]. The paper *"Vertex Analysis on Graphs"* by Shuman, Ricaud, and Vandergheynst [110] also helped frame the perspective presented here, and is an exceptional exploration into the connections between graph theory and signal processing ideas in harmonic analysis.

Geometric data science is concerned with the analysis of datasets through modeling the data as coming from a geometric domain and study the geometry. In this thesis, the assumption is that similarity between data points can be represented by a metric, often the Euclidean metric.

Datasets are typically given as data matrices; consider a matrix $X \in \mathbb{R}^{N\times d}$ of N row vectors in $\mathbb{R}^d$. In most cases, we consider d to be the number of measurements for each of the N points, but no assumption is made that would allow for the data matrix to be directly interpreted as a linear operator; the observations may be highly correlated or linearly dependent. We will abuse notation and use X to refer to both the set of data points and the matrix representation of those points and clarify when necessary.

This leaves little in the way of analyzing the data without additional contextual information, so we assume that points which are close in the Euclidean metric on $\mathbb{R}^d$ are points that share some relevant features or sense of similarity.

A fruitful idea has been to build a graph based on the dataset, as it is an object that could only carry information about the similarity between two nodes.

Definition 3.3.1. *A weighted undirected graph G is a pair $G = (S, \Sigma)$ where S is a set of nodes and $\Sigma \subseteq S \times S$ is a set of edges equipped with a weight function $w : S \times S \to \mathbb{R}$ such that*

$w(i,j) = w(j,i)$ *for all* $i, j \in S$.

Note that the unweighted graph case is the same as having the weight function $w \equiv 1$ on the entire edge set. We also require our data graphs not have any self-loops, i.e. $w(i,i) = 0$ for all $i \in S$. In this way, we could alternatively first choose a weight function satisfying the above properties and define the edge set as the support of the weight function.

A slight but important distinction to make is between the situation in which the data is *a priori* given as a graph and that when we start with a data matrix and create a weight function which is a function of the Euclidean distance between two data vectors. Here, we primarily consider the latter but the techniques we utilize are applicable in either unless otherwise stated. This difference becomes more pronounced when one wants to think about a graph as a domain on which signals live and not the signal to be analyzed itself.

While there are many ways to develop graphs from data matrices, the procedures usually follow the same few steps. We will specifically only consider weight functions which are radial basis functions because of their ubiquity in machine learning and their symmetry which naturally induces undirectedness in graphs. Given a data matrix X, let x_i be the i-th row of X. A node set is then formed by identifying vector x_i with node i for $1 \leq i \leq N$. If it makes sense for the resulting graph to be fully connected, then all that remains is to choose a function f on $\mathbb{R}$ such that $w(i,j) = f(\|x_i - x_j\|)$ for $i \neq j$ and $w(i,i) = 0$. A common choice of such a function is the Gaussian $f(r) = \exp(\frac{-r^2}{2\sigma^2})$ with some *kernel bandwidth* parameter $\sigma > 0$.

However, it is often computationally unfeasible or contextually inappropriate for every node to have a nonzero edge weight to every other node. One way of fixing this is replacing f with $f\mathbb{1}_{[0,\epsilon)}$ which only creates an edge between i and j if $\|x_i - x_j\| < \epsilon$ for some cutoff ϵ.

We will instead define for each node i, given a positive integer m, its m-nearest neighbors set $N_m(i) \subseteq S$ as the set of the m data points x_j closest to x_i, excluding x_i itself. With a radial basis function, we then define $w_i(j)$ for each node i as $f(\|x_i - x_j\|)$ for $i \in N_m(i)$ and 0 otherwise. Our weight function then is just $w(i,j) = \max\{w_i(j), w_j(i)\}$. The nearest neighbors method is often chosen over the ϵ-neighborhood because with nearest neighbors, it's easier to control the number of nodes connected to each node. Also, due to tree search algorithms for finding pairwise distances of points, the nearest neighbors method has a lower computational complexity as one can avoid computing the full set of pairwise distances [56, 102].

The resultant graph $G = G(X) = (X, \Sigma(X))$ is then something that can be analyzed using the tools of spectral graph theory. The theory essentially studies graphs based on the spectral properties of matrices associated to the graph.

Definition 3.3.2. *For a graph G, define the adjacency matrix $A \in \mathbb{R}^{N\times N}$ by $A_{ij} = w(i,j)$.*

By construction of our weight function, A is a real symmetric matrix with 0 down its diagonal. If we further impose that the weight function is nonnegative, e.g. if we choose the Gaussian, then the adjacency matrix is symmetric positive semi-definite and thus has real nonnegative eigenvalues. For the rest of this text, we only consider nonnegative weight functions. We can thus establish the notation of $i \sim j$ to represent the symmetric relation $i \sim j \iff w(i,j) > 0$.

Definition 3.3.3. *The degree matrix $D \in \mathbb{R}^{N\times N}$ for a graph G is the diagonal matrix where*

$$D_{ii} = \sum_{j=1}^{N} w(i,j).$$

In the case of an unweighted graph, the degree at every node is just the number of edges

incident to it.

Definition 3.3.4. *The graph Laplacian $\mathcal{L}$ of a graph G is the difference between its degree and weight matrices, $\mathcal{L} = D - A$.*

The graph Laplacian and its following normalization make up the primary object of study in the analysis of graph signals detailed in this paper, although spectral properties of the adjacency matrix are often used as well, particularly when considering random walks and diffusion-like phenomena on graphs.

Definition 3.3.5. *The normalized graph Laplacian L is given by*

$$L = D^{-\frac{1}{2}}\mathcal{L}D^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}.$$

The normalized version of the Laplacian has some nice properties, including that its spectrum is contained in $[0, 2]$. This can be seen by examining the similar normalized adjacency operator $A_N = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$. A_N is positive semidefinite as all of its entries are nonnegative. Hence, $I + A_N$ is also positive semidefinite, with identical eigenvectors as A_N. Then the following holds for all $v \in \mathbb{R}^N$:

$$\begin{aligned} v^\top (I + A_N)\, v \geq 0 \implies v^\top v + v^\top A_N v \geq 0 \implies \\ v^\top v \geq -v^\top A_N v \implies 1 \geq \frac{-v^\top A_N v}{v^\top v} \implies \\ -1 \leq \frac{v^\top A_N v}{v^\top v}. \end{aligned}$$

This means that $-1 \leq \min_v \frac{v^\top(-A_N)v}{v^\top v}$ and so the smallest eigenvalue of A_N is at least -1. Every eigenvector of A_N is trivially also one of both I with eigenvalue 1, and $-A_N$ with opposite

eigenvalue. As a result, the largest eigenvalue of $-A_N$ is at most 1, and the largest eigenvalue of $I - A_N = L$ is at most 2.

An alternative formation of the theory is to consider matrices A, D, and L as *linear operators* acting on the Hilbert space $V(G)$ of real-valued functions on G. Let X be a finite or countably infinite set of nodes with a symmetric, nonnegative weight function w. In this more general case, we impose the light restriction that $\sum_{x_j \in X} w_i(j) < \infty$ for all $x_i \in X$. Then the adjacency operator A acts on a function $v \in V(G)$ by

$$Av(i) = \sum_{x_j \in X} w(i,j)v(j).$$

As w is real-valued and symmetric, A is self-adjoint.

This means that A acts on a function v at x_i by creating a weighted sum of the values of v in the edge set of x_i. Similarly, we define the degree operator D as

$$Dv(i) = d(i)v(i) = \left(\sum_{x_j \in X} w(i,j) \right) v(i).$$

The operator D acts as a pointwise multiplier of the degree of each node. For this, we add the requirement that $d(i) < \infty$ for each x_i. Then finally, we define the normalized graph Laplacian operator L by

$$Lv(i) = \frac{1}{\sqrt{d(i)}} \left(v(i)\sqrt{d(i)} - \sum_{x_j \in X} w(i,j) \frac{v(j)}{\sqrt{d(j)}} \right) = v(i) - \sum_{x_j \in X} w(i,j) \frac{v(j)}{\sqrt{d(i)d(j)}}.$$

The Laplacian operator L on v at x_i measures the difference between $v(i)$ and a weighted and degree-normalized average of the values of v in the edge set of x_i. In this way, we can note that

L captures exactly when a function v exhibits a *mean value property*; that is, $Lv = 0$ means that $v(i) = \sum_{x_j \in X}$ for all $x_i \in X$. Note further that this property extends for any eigenfunction of L. If (λ, v) is an eigenpair for L, then

$$v(i) - \sum_{x_j \in X} w(i,j) \frac{v(j)}{\sqrt{d(i)d(j)}} = \lambda v(i) \implies (1-\lambda)v(i) = \sum_{x_j \in X} w(i,j) \frac{v(j)}{\sqrt{d(i)d(j)}}.$$

This operator characterizes a notion of *smoothness* of functions on the graph.

Referring to the operator as a *Laplacian* is motivated by analogy to the second finite difference operator on a grid, and it is indeed a discrete version of the Laplace-Beltrami operator under certain conditions [12].

Throughout this thesis, all graphs will be assumed to have a finite set of nodes. However, this assumption is not strictly required for developing the theory of graph spectral analysis. In such situations, extra care is needed to deal with the convergence of sums required for defining the graph operators. Infinite graphs do lose some of the properties which make the theory applicable to data analysis. For example, the kernel of the Laplacian operator of an infinite matrix does *not* describe the number of connected components of the graph like in the finite case. For example, consider the unweighted graph on $\mathbb{Z}$. Here, $w(i,j) = 1 \iff |i-j| = 1$ and $w(i,j) = 0$ otherwise. Clearly, $d(i) = 2$ for all $i \in \mathbb{Z}$. Take v as an arithmetic progression $v(i) = ai + b$ for some $a, b \in \mathbb{R}$. Then

$$(Lv)(i) = v(i) - \left(\frac{v(i-1)}{2} + \frac{v(i+1)}{2} \right) = v(i) - \left(\frac{v(i)}{2} - \frac{a}{2} + \frac{v(i)}{2} + \frac{a}{2} \right) = v(i) - v(i) = 0.$$

Therefore, all arithmetic progressions are annihilated by the Laplacian, and the space of

such maps is 2-dimensional, yet the graph is clearly connected. Omissions like this make the application of the infinite graphs more difficult to conceive. More traditional Fourier analysis is likely sufficient for systems with enough regular structure.

3.3.1 Laplacian Eigenmaps

From the previous section, we see that eigenvectors of the smallest d nonzero eigenvalues, $\{\phi_1, \ldots, \phi_d\}$, are orthogonal and form a basis for the d-dimensional subspace of $V(G)$ consisting of the smoothest functions. Projecting a function onto this space along these eigenfunctions gives a smooth approximation to the function. Let $\delta_i \in V(G)$ such that $\delta_i(x_i) = 1$ and $\delta_i = 0$ otherwise. This function can be thought of as representing evaluation at x_i, in the sense that $\langle v, \delta_i \rangle = v(i)$ for all $v \in V(G)$. The projection of this function into the span of the smallest d nonzero eigenvectors is then a smooth approximation of the evaluation function. The coordinates of this projection also can be considered as a set of coordinates for x_i in $\mathbb{R}^d$ which encapsulate similarity of nodes in terms of Euclidean distance.

Definition 3.3.6. *Let $G = (X, \Sigma(X))$ be a finite weighted undirected graph with $|X| = n$ and normalized graph Laplacian L. Denote the eigenfunctions of the smallest d nonzero eigenvalues as $\{\phi_\ell\}_{\ell=1}^n$. Then the (d-dimensional) Laplacian eigenmaps embedding, or Laplacian embedding, $\Phi : X \to \mathbb{R}^d$ is defined as*

$$\Phi(x_i) = (\langle \delta_i, \phi_\ell \rangle)_{\ell=1}^d = (\phi_1(i), \phi_2(i), \ldots, \phi_d(i)).$$

In an abuse of notation, we will denote by Φ both the Laplacian eigenmaps embedding *and* the matrix $[\phi_1 \cdots \phi_d] \in \mathbb{R}^{n \times d}$ consisting of the eigenfunctions of the smallest d eigenvalues

which define the embedding. That is, we should not think of the matrix Φ as having the full set of eigenfunctions of the normalized graph Laplacian, nor should we consider the operator Φ as having a matrix representation given by the matrix Φ.

3.4 Creating a Graph from an Image

We start by considering an n_1 by n_2 image $\mathcal{P}$ as a set of $n = n_1 \cdot n_2$ pixels, enumerated in some way $\mathcal{P} = \{p_i\}_{i=1}^n$. A digital image provides an interesting case for data analysis using graph methods because there is already a natural grid structure. Natural images are known to be highly spatially correlated, so it is very reasonable to create a graph on an image in which the edges connect adjacent pixels. However, we first disregard this more apparent graph structure. Our goal is to create a graph which encodes the pixel value similarity. At a later point, we can incorporate the spatial content of the image if desired. Typical images have pixels which are represented as a combination of red, green, and blue values. In this way, we can consider the set of pixels $\{p_i\}_{i=1}^n$ as a subset of the unit cube $[0, 1]^3$. We imbue the Euclidean distance onto the set of pixels in order to capture a measure of color similarity. In the case of grayscale images, where pixels only capture the intensity of light information, the set $\{p_i\}_{i=1}^n$ is a subset of the unit interval $[0, 1]$. This set of pixel vectors will form the nodes of the image graph. In subsequent sections, we will explore alternative methods for constructing a set a pixel vectors.

Next, we construct the weight function which uses a nearest neighbors condition. Let $N_k(i)$ be the k-neighborhood set for pixel p_i. Then, for each such p_i, we define a local weight function in the way illustrated earlier, $w_i : \mathcal{P} \to \mathbb{R}$ as $w_i(j) = \exp(\frac{\|p_i - p_j\|_2^2}{2\sigma^2})$ for $p_j \in N_k(i)$ and $w_i(j) = 0$ otherwise. In particular, we have that $w_i(i) = 0$ since p_i is excluded from its neighbor

set. The weight function for the graph is then defined as $w(i,j) = \max\{w_i(j), w_j(i)\}$. Defining the weight function as a maximum ensures that the function is symmetric, but this creates the possibility that $w(i,j) > 0$ while $j \notin N_k(i)$ as long as $i \in N_k(j)$.

The graph $G_{\mathcal{P}} = (\mathcal{P}, \Sigma(\mathcal{P}))$ is then constructed with $(p_i, p_j) \in \Sigma(\mathcal{P})$ if and only if $w(i,j)$. We form an adjacency matrix A as $a_{ij} = w(i,j)$. This allows for defines the degree matrix D and the normalized graph Laplacian L as described earlier.

3.5 Out of Sample Extension via Nyström Approximation

The Laplacian eigenmaps embedding is wholly determined by the relationships within the data, not necessarily the original representation of data points in the ambient space. While this makes the method adaptive to data that has very little observable structure, a drawback of this approach is that the embedding is not defined for points outside that data set. This begs the question of how one deals with other points in the ambient data space. One method of approximating the embedding of some new, out-of-sample point $y \in \mathbb{R}^D$ is to use the method outlined by Bengio et al. [18] which is based on the Nyström method.

To illustrate this method, first let $X = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^{n \times D}$ be a data set in the form of a matrix, with data points as rows. By choosing some weight function w, we can form the Laplacian eigenmaps embedding $\Phi : X \to \mathbb{R}^d$ as described in Section 3.3.1. We can identify $\Phi = \{\phi_\ell\}_{\ell=1}^d$ with the eigenfunctions of the d-smallest nonzero eigenvectors of the graph Laplacian, since those eigenfunctions generate the embedding. Then, as an eigenfunction, $L\phi_\ell(i) = \lambda_\ell \phi_\ell(i)$. But from the definition of the graph Laplacian, $L\phi_\ell(i) = \phi_\ell(i) - \sum_{x_j \in X} w(i,j) \frac{\phi_\ell(j)}{\sqrt{d(i)d(j)}}$. Combining those two equations, we can relate the ℓ-th coordinate of the embedding of x_i as a

function of the same coordinates of the neighbors of x_i:

$$\lambda_\ell \phi_\ell(i) = \phi_\ell(i) - \sum_{x_j \in X} w(i,j) \frac{\phi_\ell(j)}{\sqrt{d(i)d(j)}} \implies (1-\lambda_\ell)\phi_\ell(x_i) = \sum_{x_j \in X} w(i,j) \frac{\phi_\ell(j)}{\sqrt{d(i)d(j)}}$$

$$\therefore \phi_\ell(x_j) = \frac{1}{1-\lambda_\ell} \sum_{x_j \in X} w(i,j) \frac{\phi_\ell(j)}{\sqrt{d(i)d(j)}}.$$

We can write this as a matrix vector product:

$$\Phi(x_i) = K_{x_i} \Phi (I - \Lambda)^{-1}$$

where K_{x_i} is the row vector such that $(K_{x_i})_j = \frac{w(i,j)}{\sqrt{d(i)d(j)}}$. In this case, we refer to K_{x_i} as the *kernel vector* of x_i. This alternate expression inspires a way to extend the LE embedding to other points in the ambient space. For some point $y \in \mathbb{R}^D$, not necessarily in X, we can define an extended map as long as we are able to form the set of nearest neighbors in X of y.

Definition 3.5.1. *The Nyström extension $\widetilde{\Phi} : \mathbb{R}^D \to \mathbb{R}^d$ of the Laplacian embedding Φ is defined as*

$$\widetilde{\Phi}(y) = \sum_{\ell=1}^{d} \frac{1}{1-\lambda_\ell} \sum_{x_j \in \mathcal{N}_y} \tilde{w}(y, x_j) \frac{\phi_\ell(j)}{\sqrt{\tilde{d}(y) d(x_j)}}$$

where $\mathcal{N}_y \subseteq X$ is the set of nearest neighbors to y in X, $\tilde{w} : \mathbb{R}^D \times X \to \mathbb{R}$ is an extended weight function, and $\tilde{d}(y)$ is the sum of $\tilde{w}(y, x_j)$ over all $x_j \in \mathcal{N}_y$.

3.6 L^1 Regularized Preimage for Laplacian Eigenmaps

In this section, we describe the technique developed by Cloninger, Czaja, and Doster [31] for approximating a preimage to the Laplacian Eigenmap embedding. The sparsity provided

by the nearest neighbor condition of LE serves a similar function in this preimage method as a regularizer and as way to reduce computational cost. The approximate preimage method has three major components: approximating a kernel vector, creating a neighborhood set and defining distances, and recovering coordinates from distances via MDS.

Our goal in this section is to find, given some point ψ in the Laplacian embedding space, a point y in the ambient data space whose embedding would approximate ψ. This procedure is summarized in Algorithms 3.1, 3.2, and 3.3.

3.6.1 Approximating the Kernel Vector

Begin with the same construction of a Laplacian embedding from a set $X \in \mathbb{R}^{n\times D}$ as in Section 3.5, given by $\Phi : X \to \mathbb{R}^d$. This induces a Nyström extension

$$\widetilde{\Phi}(y) = \sum_{\ell=1}^{d} \frac{1}{1-\lambda_\ell} \sum_{x_j \in \mathcal{N}_y} \tilde{w}(y, x_j) \frac{\phi_\ell(j)}{\sqrt{\tilde{d}(y) d(x_j)}}$$

for some point $y \in \mathbb{R}^D$. Suppose then that there is some point $\psi \in \mathbb{R}^d$ which is in the embedding space containing the Laplacian embedding. That is, ψ is a point in the codomain of the embedding (or extension) map but is not necessarily the image of any in-sample point $x \in X$ used to create the embedding. Clearly, as the definition of the Laplacian embedding at each point requires information for *all* other points in the sample set, we cannot expect the embedding to be surjective. However, under the mild assumption that the set of embedded sample points $\Phi(X)$ forms a spanning set, then we can express ψ as a linear combination of embedded points, $\psi = \sum_{i=1}^{n} \psi_n \Phi(x_i)$. Recall from Section 3.3.1 that $\Phi(x_i)$ is also given by the i-th row of the matrix Φ of the first d eigenvalues of the normalized graph Laplacian. We will then use $\Phi(i)$ to denote the i-th row of

Φ. This means that ψ is expressible as the linear combination $\psi = \sum_{i=1}^{n} \psi_i \Phi(i)$, which is the matrix-vector product $\psi = v\Phi$ for $v = [\psi_1, \ldots, \psi_n] \in \mathbb{R}^n$.

If we require that ψ be the image of some $y \in \mathbb{R}^D$ under the extension $\widetilde{\Phi}$, then we know that the weights of the linear combination of embedded sample points must somehow reflect the relationship between y and each data point $x_i \in X$. This specific matrix-vector product is expressed as follows:

$$\psi = \widetilde{\Phi}(y) = \sum_{\ell=1}^{d} \frac{1}{1-\lambda_\ell} \sum_{x_j \in \mathcal{N}_y} \tilde{w}(y, x_j) \frac{\phi_\ell(j)}{\sqrt{\tilde{d}(y) d(x_j)}} = \sum_{x_j \in \mathcal{N}_y} \sum_{\ell=1}^{d} \frac{\tilde{w}(y, x_j)}{\sqrt{\tilde{d}(y) d(x_j)}} \frac{1}{1-\lambda_\ell} \phi_\ell(j) =$$

$$= \sum_{x_j \in \mathcal{N}_y} \frac{\tilde{w}(y, x_j)}{\sqrt{\tilde{d}(y) d(x_j)}} (\Phi(j)(I_d - \Lambda)^{-1}) = k_y \Phi (I_d - \Lambda)^{-1}$$

where $k_y \in \mathbb{R}^n$ is the row vector whose j-th entry is given by $(k_y)_j = \frac{\tilde{w}(y, x_j)}{\sqrt{\tilde{d}(y) d(x_j)}}$ for $x_j \in \mathcal{N}_y$ and $(k_y)_j = 0$ for $x_j \notin \mathcal{N}_y$ and $\Lambda \in \mathbb{R}^{d \times d}$ is the diagonal matrix of eigenvalues of the Laplacian matrix. Thus, in order to recover y from $\psi = \widetilde{\Phi}(y)$, we need to first solve for this kernel vector k_y. However, rather than directly solving the least squares problem $\widehat{k_y} = \operatorname{argmin}_{K \in \mathbb{R}^n} \|K\Phi - \psi\|_2$, we can utilize the sparsity condition of the Laplacian embedding, and correspondingly, its Nyström extension, as both a regularizer and a tool with which to learn the neighborhood set $\mathcal{N}_y$.

In order to develop how the nearest neighbor condition allows for computation of the kernel vector, we will focus on the case of trying to learn the kernel vector for an in-sample point, which we denote as $\Phi(x_i)$. Recall that the only nonzero adjacencies $w(i, j)$ are for points x_j which are among the m-nearest neighbors of x_i, and that those points form the neighborhood set $\mathcal{N}_m(i)$. In

this case, we have

$$\Phi(x_i)(I_d - \Lambda) = \sum_{\ell=1}^{d} \sum_{x_j \in \mathcal{N}_{x_i}} \frac{w(i,j)}{\sqrt{d(i)d(j)}} \phi_\ell(j) = k_{x_i}\Phi.$$

We can then take m as an approximation of the ℓ_0 norm of k_{x_i}, while acknowledging that m is only a lower bound of the set of x_j such that $i \sim j$.

Our goal here is to translate the ℓ^0 norm condition $\|k_{x_i}\|_0 < m$ into an approximate ℓ^1 condition $\|k_{x_i}\|_0 < \tau$, with the temporary assumption that $i \sim j \implies x_j \in \mathcal{N}(i)$. In such a case, we can say that

$$\|\psi\|_2 = \|k_{x_i}\Phi\|_2 \implies \frac{\|\psi\|_2}{\|k_{x_i}\|_1} = \frac{1}{\|k_{x_i}\|_1}\|k_{x_i}\Phi\|_2 = \|\frac{k_{x_i}}{\|k_{x_i}\|_1}\Phi\|_2 =$$

$$= \|\sum_{\ell=1}^{d} \sum_{x_j \in \mathcal{N}(i)} a_j \phi_\ell(j)\|_2$$

where $a_j = \frac{(k_{x_i})_j}{\|k_{x_i}\|_1}$. We then use the SVD of the matrix Φ, given by $\Phi = U\Sigma V^\top$, to compute its *pseudoinverse* $\Phi^\dagger = V\Sigma^\dagger U^\top$, where $\Sigma^\dagger$ is the matrix in which the nonzero values of $\Sigma^\top$ are replaced with their reciprocals. Let $K = \psi\Phi^\dagger = k_{x_i}\Phi\Phi^\dagger$. The m largest values of K, $\{K_{i_1}, \ldots, K_{i_m}\}$, are then chosen to represent the m-largest values of k_{x_i}, which should be exactly those corresponding to weights between neighboring points. Let $\hat{a_{i_s}} = \frac{K_{i_s}}{\sum_s K_{i_s}}$, then we define $\hat{\alpha}$ by

$$\hat{\alpha} = \|\sum_{\ell=1}^{d} \sum_{s=1}^{m} \hat{a_{i_s}} \phi_\ell(x_{i_s})\|_2.$$

This gives us $\hat{\alpha}$ as an approximation of the ratio $\frac{\|\psi\|_2}{\|k_{x_i}\|_1}$ which in turn means that $\|k_{x_i}\|_1$ can be approximated by $\frac{\|\psi\|_2}{\hat{\alpha}}$, a quantity that can be calculated using only knowledge of the embedded

point ψ. Therefore, we can set $\tau \geq \frac{\|\psi\|_2}{\hat{\alpha}}$ to recover an approximate ℓ^1-norm bound from the nearest neighbor condition.

Thus, the first step in computing the preimage y of a point ψ is the solving the convex optimization problem

$$\widehat{k_y} = \underset{\substack{k \in \mathbb{R}^n \\ \|k\|_1 \leq \tau}}{\operatorname{argmin}} \|k\Phi - \psi\|_2.$$

Then, we set all but the m-largest values of $\widehat{k_y}$ equal to 0, ensuring that $\|\widehat{k_y}\|_0 = m$.

We solve the optimization using a spectral projected gradient method implemented in MATLAB. Note that since the weight function is nonnegative, the kernel vector k_y should also have a positivity constraint. In practice, we found that this added complexity to the overall method, as the kernel vector search is the computational bottleneck of the preimage algorithm. We addressed this issue by simply initializing the iterative method by setting the initial solution to a projection of $\psi\Phi^\dagger$ into the positive half-space $\mathcal{H}^n$. This can be done with component-wise application of the linear rectifier function $ReLU(x) = max(0, x)$. Although this does not guarantee positivity of the resultant solution, we found that this more consistently lead to a positive solution when compared to initializing with the zero vector. In any case, since we only choose the largest m values of the kernel vector moving forward, this heuristic is usually good enough.

Algorithm 3.1 Approximating the kernel vector $\widehat{k_y}$

Require: Φ, ψ, m
Compute pseudoinverse $\Phi^\dagger$
$K \leftarrow \psi\Phi^\dagger$
i_s is the index of the s-largest value of K.
$\widehat{a_{i_s}} \leftarrow K_{i_s} / \sum_r^m K_{i_r}$
$\hat{\alpha} \leftarrow \|\sum_{\ell=1}^d \sum_{s=1}^m \hat{a_{i_s}} \phi_\ell(x_{i_s})\|_2$
$\tau \leftarrow \|\psi\|_2 / \hat{\alpha}$
Initialize $k_0 \leftarrow ReLU(K)$
return $\widehat{k_y} = argmin_{k \in \mathbb{R}^n} \|k\Phi - \psi\|_2$ subject to $\|k\|_1 < \tau$

3.6.2 Computing Neighborhood Distance

The next step in calculating the approximate preimage is recovering neighborhood distances from the kernel vector found in the previous step. These distance can be solved for using knowledge of the degrees of the in-sample points.

Let us again look first to the recovery of the preimage of an in-sample point x_i from its embedding $\psi = \Phi(x_i) = k_{x_i}\Phi$. From the approximated kernel vector $\hat{k}$ computed in the previous section, we use $\{i_1, \ldots, i_m\}$ to denote the indices corresponding to the nonzero values of $\hat{k}$. For $1 \leq s \leq m$, define e_s as the value of the weight function between x_i and x_{i_s}, that is, $e_s = w(x_i, x_{i_s})$. That means that the values of the kernel vector can be written as

$$(\hat{k})_{i_s} = \frac{e_s}{\sqrt{\sum_r e_r \sum_{x_j \sim x_{i_s}} w(x_{i_s}, x_j)}}$$

So, for each s, we define b_s as

$$b_s = \frac{e_s}{\sqrt{\sum_r e_r}} = (\hat{k})_{i_s} \sqrt{\sum_{x_j \sim x_{i_s}} w(x_{i_s}, x_j)}.$$

We can think of each b_s as an edge weight of x_i, normalized by the degree of x_i. Note that each b_s term has the same denominator which is $\sqrt{d(i)} = \sqrt{\sum_s e_s}$. In the general case however, we may not be able to calculate the degree of some generic preimage which may not be part of the in-sample set. We do have access to the degrees of the x_{i_s} in-sample points though, and those are used to ultimately recover the weights out of the entries of $\hat{k}$. As each

$b_s = (\hat{k})_{i_s}\sqrt{\sum_{x_j \sim x_{i_s}} w(x_{i_s}, x_j)} = (\hat{k})_{i_s}\sqrt{d(i_s)}$, we know that

$$\sum_s b_s = \frac{\sum_s e_s}{\sqrt{\sum_s e_s}} = \sqrt{\sum_s e_s}$$

can be computed as long as we know the degrees of each x_{i_j}. Then $B = \sum_s b_s$ clears the denominator of b_s which gives $e_s = Bb_s$ for each s, and thus we've calculated the approximate weights for each of the proposed neighbors of x_i.

Therefore, in the general case of preimage y and embedded point ψ, we can compute e_s for $1 \leq s \leq m$ as above and define the neighborhood set of y as $\mathcal{N}_y = \{x_{y_1}, \ldots, x_{y_s}\}$. The neighborhood distances are then given by $\|y - x_{y_s}\|_2^2 = -2\sigma^2 \log(e_s)$.

Algorithm 3.2 Calculating neighborhood distances, following Algorithm 3.1

Require: A, X, $\widehat{k_y}$, m
  i_s is the index of the s-largest value of $\widehat{k_y}$.
  $b_s \leftarrow \left(\widehat{k_y}\right)\sqrt{\sum_{x_j \sim x_{i_s}} A_{i_s j}}$
  $B \leftarrow \sum_s^m b_s$
  $e_s \leftarrow B \cdot b_s$
  $\mathcal{N}_y \leftarrow \{x_{i_s}\}_{s=1}^m$
  return Distance from $\hat{y}$ to x_{i_s} is $-2\sigma^2 \log(e_s)$

3.6.3 Recovering Coordinates in the Ambient Data Space

The final step in the preimage calculation is to recover the ambient space coordinates of $\hat{y}$ using the squared distances. This is done as a least squares problem and comes from a strategy for inverting kernel PCA which is based on MDS [73].

We start with defining the centering matrix $H \in \mathbb{R}^{m \times D}$ as $H = I_m - \frac{1}{m}\mathbb{1}\mathbb{1}^\top$ with I_m as the $m \times m$ identity matrix and $\mathbb{1} \in \mathbb{R}^m$ as the column vector of all 1s. In the previous step, we

defined the neighborhood set $\mathcal{N}_y$ for the preimage y that we wish to find. Store the coordinate vectors of the neighborhood points as rows in the matrix $X_0 = [x_{y_1}^\top, \ldots, x_{y_m}^\top]^\top \in \mathbb{R}^{m \times D}$. Then the product HX_0 is the centered version of X_0 in which mean over all rows of X_0, denoted as $\bar{x}$, is subtracted off of each row.

Our goal is to solve for y as the point for which the distances to X_0 are as close to the distances computed in the previous section as possible. We can do this by first taking the SVD of $HX_0 = U\Sigma V^\top = ZV^\top$ where $Z = U\Sigma$. By orthogonality of V, multiplying gives us $HX_0V = Z$, which means Z is the projection of the centered neighborhood points into the column space of V. Importantly, the mutual distances between the points, as well as their norms, are preserved by multiplication from both H and V. Let $\hat{z}$ be the projection into columns of V of the centered version of the preimage y, so $\hat{z} = (y - \bar{x})V$. In a perfect reconstruction, we would have $\|\hat{z} - z_{y_s}\|_2 = \|y - x_{y_s}\|_2$ for all $1 \leq s \leq m$. Then let $r = [\|x_{y_1}\|_2^2, \ldots, \|x_{y_m}\|_2^2]$ be the row vector of squared norms of the neighborhood points and r_0 be the row vectors of squared distances computed from the kernel vector in the previous section. By taking a difference of the two squares and then centering, we can see that $(r - r_0)H = -2\hat{z}Z^\top$. This means we have $\hat{z}$ as the least squares solution $\hat{z} = -\frac{1}{2}(r - r_0)HZ(Z^\top Z)^{-1}$. Note also that $HZ = H(HX_0V) = H^2X_0V = HX_0V = Z$, because the centered version of X_0 has mean 0, i.e. $H^2 = H$. Thus, we can calculate

$$\hat{z} = -\frac{1}{2}(r - r_0)HZ(Z^\top Z)^{-1} = -\frac{1}{2}(r - r_0)Z(Z^\top Z)^{-1} = -\frac{1}{2}(r - r_0)Z(\Sigma^\top U^\top U\Sigma)^{-1} =$$

$$= -\frac{1}{2}(r - r_0)Z(\Sigma^\dagger)^2 = -\frac{1}{2}(r - r_0)(Z\Sigma^\dagger)\Sigma^\dagger = -\frac{1}{2}(r - r_0)U\Sigma^\dagger.$$

Finally, we obtain the approximate preimage $\hat{y}$ of ψ by projecting back and adding the mean: $\hat{y} = \hat{z}V^\top + \bar{x}$.

Note that for data with some regular structure, it is not unreasonable to have a constant set of approximated neighbors. In such cases, this final calculation cannot be completed because of the degeneracy of the SVD. However, this situation is easily remedied by artificially setting the preimage to that constant approximated value.

Algorithm 3.3 Recovering preimage $\widehat{y}$ using MDS, following Algorithm 3.2

Require: $X, e_1, \ldots, e_m, m, i_1, \ldots, i_m$
  X_0 is the submatrix of rows $i_1, \ldots, i_m$ of X
  Compute SVD of HX_0, $HX_0 = U\Sigma V^\top$
  $\bar{x}$ is the centroid of points $x_{i_1}, \ldots, x_{i_m}$
  $r \leftarrow [\|x_{i_1}\|_2^2, \ldots, \|x_{i_m}\|_2^2]$
  $r_0 \leftarrow [-2\sigma^2 \log(e_1), \ldots, -2\sigma^2 \log(e_m)]$
  if $x_{i_1} = \cdots = x_{i_m} = \bar{x}$ **then**
    $\widehat{y} \leftarrow \bar{x}$
  else
    $\widehat{y} \leftarrow \bar{x} - \frac{1}{2}(r - r_0)U\Sigma^\dagger V^\top$
  end if
  return $\widehat{y}$

Chapter 4: Main Contributions

We begin this chapter with a presentation of our novel curvature-based clustering method that is designed for combination with Laplacian eigenmap (LE) embeddings. Next, we develop two methods for mapping between LE embeddings given some initial known correspondence. Then, we present our Laplacian eigenmaps data fusion model. We conclude with an exploration of problems in MR image processing and supervised learning on genomics data which can be solved within this data fusion framework.

4.1 Curvature-Based Clustering

In this section, we describe a clustering algorithm for a point set in $\mathbb{R}^2$ which is designed to create clusters which exhibit *flatness*, in the sense that the shortest path between two points in a cluster is close to their Euclidean distance in the ambient space. This can be seen as a generalization of the concept of a *Riemannian metric* on a smooth manifold embedded in Euclidean space; in this sense, *flatness* is characterized by near-equality of geodesic distance on the manifold and the Euclidean distance imbued through its embedding. This idea is particularly useful in the case of Laplacian embedded points, as the overall geometric structure of the embedding encodes information about a data set.

Suppose that $X \subset \mathbb{R}^2$ is a finite set of points. To start, we first construct the Delaunay

triangulation of the set to obtain a surface. We then want to define a boundary for this surface. As we have no geometric or convexity assumptions on the set, we need to specify the notion of the boundary of this surface. For this algorithm, we choose to define the boundary as an α-surface, although this choice is not essential. In general, the boundary should be a curve which is completely contained inside the convex hull of the set.

Consider the Delaunay triangulation as a pair consisting of the set X and an edge set $\mathcal{E} \subseteq X \times X$. The α-shape requires the notion of a *generalized disk*, a closed, connected set in the plane which allows for a radius to take any real value. For $\alpha > 0$, the generalized disk of radius $\frac{1}{\alpha}$ coincides with the usual notion of a disk of radius $\frac{1}{\alpha}$ in $\mathbb{R}^2$. For $\alpha = 0$, the generalized disk of radius $\frac{1}{\alpha}$ is a closed half-plane, and for $\alpha < 0$, the generalized disk of radius $\frac{1}{\alpha}$ is the closure of the complement of the closed disk of radius $-\frac{1}{\alpha}$.

Then the *α-shape* for the Delaunay triangulation $(X, \mathcal{E})$ is the subset $\mathcal{E}_\alpha \subseteq \mathcal{E}$ of the edge set consisting of edges for which a generalized disk of radius at most $\frac{1}{\alpha}$ has the boundary of the edge on its boundary, and the interior of this generalized disk has empty intersection with the set X. This can be though of as a generalization of the convex hull of a point set, as the convex hull of X is realized as its α-shape with $\alpha = 0$. We then choose the boundary curve $\mathcal{C}$ as the union of the set of all edges in $\mathcal{E}_\alpha$ which lie on only one face of the Delaunay triangulation.

If the set X happens to be convex, then we can simply set $\alpha = 0$, in which case the boundary curve would be the convex hull. Otherwise, α is a parameter which is chosen to preserve a sense of the shape of the set. A reasonable heuristic would be to set α as some function of the average distance between points which share an edge in $\mathcal{E}$. In our implementation for a Laplacian embedding which is restricted to $[-1, 1]^2$, setting $\alpha = 0.5$ seems to be sufficent for creating a closed boundary curve which fits the embedded points.

Thus, $\mathcal{C}$ is a piecewise linear closed curve, and we proceed by calculating its curvature. This can be done by using the characterization of the curvature of the curve $\gamma : [0,1] \to \mathbb{R}^d$ at a point $\gamma(t)$ as the inverse of the radius of its osculating circle at that point. For our piecewise linear curve, we instead use a finite approximation of this. To calculate the curvature $\kappa(i)$ at a point $\mathcal{C}(i)$, we use the circumscribed circle of the triangle formed by $\mathcal{C}(i-1), \mathcal{C}(i), \mathcal{C}(i+1)$ as the analogue to the osculating circle and denote its radius as $r(\mathcal{C}(i))$. Then $\kappa(i) = r(\mathcal{C}(i))^{-1}$.

Given a predetermined number of desired clusters S, we find the S points of $\mathcal{C}$ which are local curvature maximizers. These points are found by finding points on the boundary curve for which the forward second difference $\kappa(i+2) - 2\kappa(i+1) + \kappa(i)$ is negative and the sign of its forward difference changes, i.e. $(\kappa(i+2) - \kappa(i+1))(\kappa(i+1) - \kappa(i)) < 0$. In analogy to the case of differentiable functions of a real single variable, this is done to detect zero crossings of the derivative which correspond to local curvature maxima. If there are more that S such points, then we choose the S such points of largest curvature.

Denote these points as $\{c_{M_1}, \ldots, c_{M_S}\}$, ordered such that no other maximizers lie on the path on $\mathcal{C}$ from c_{M_j} and $c_{M_{j+1}}$ for all $1 \leq j \leq S$. Assign c_{M_j} to cluster j, as well as all of the points on $\mathcal{C}$ which lie between c_{M_j} and $c_{M_{j+1}}$, for each $1 \leq j \leq S$. Once all of $\mathcal{C}$ is clustered, we then cluster each of the remaining points in X by choosing for some point $x \in X$ the most frequent cluster among some the m-closest points on $\mathcal{C}$ to x. In the case of a tie, we choose the cluster of the nearest boundary point to x.

The driving feature of how this method differs from other geometry-based clustering methods such as k-means can be illustrated in the following example. Suppose X consists of regularly spaced points on some part of the parabola $y = x^2$ which contains the vertex at the origin. If we intend to cluster X into 2 parts, then the curvature-based clustering method described here would

choose a point close to the vertex as one of the curvature maximizers. This would have the effect of clustering by splitting the two sides of the parabola. On the other hand, a k-means cluster which contains a point close to the origin is likely to include a significant amount of points on both sides of the vertex.

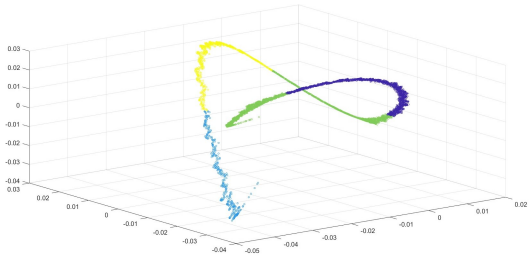
As an example, Figure 4.1 shows two different clusterings performed on the same point cloud in $\mathbb{R}^3$, designed to look like a curve with multiple turns. Seeking 4 clusters from both methods, we compare k-means clustering, implemented using the k-means++ algorithm in MATLAB [3], and the curvature-based clustering described in this section. Our clustering method is designed to partition the set into patches which are approximately linear. Notice in particular in Figures 4.1b and 4.1d that the k-means clustering includes in the same cluster two disconnected sections of the curve. The curvature clustering does a better job of respecting the metric on the curve itself, not just in the ambient space.

4.2 Mappings of LE Embeddings

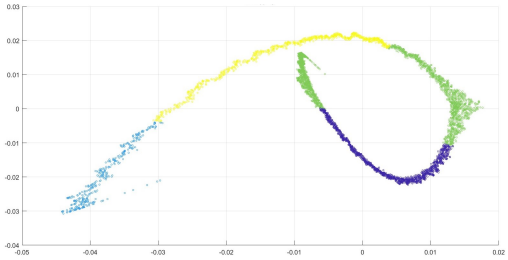
4.2.1 Rotations

Laplacian embeddings provide useful representations for dimension reduction and visualization, but we often want to compare similarly structured data across different embeddings. For instance, if we would naively suspect that a particular data set does not vary much over time, then it stands to reason that there should be some comparability between the LE embeddings of the data set at different time points. This requires some choice of a map from one embedding space to another. It is imperative that such a function should respect the neighborhood structure of the points on a reasonably small scale. This is because the entire foundational concept behind the

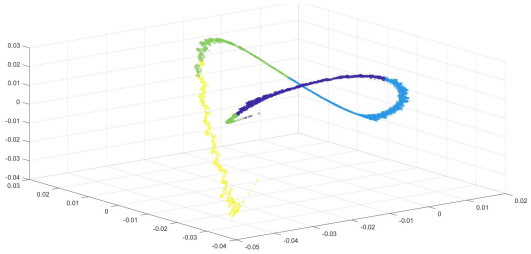
Figure 4.1: *A toy example comparing k-means and curvature-based clustering methods for Laplacian eigenmap embeddings. The k-means clustering in (a), (b) form clusters which do not split at turning points, while the curvature-based clustering in (c), (d) creates clusters which are closer to being piecewise linear.*



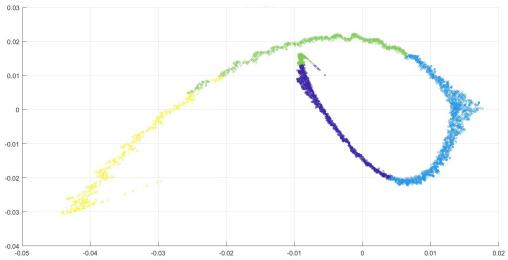
(a) k-means clustering



(b) k-means clustering, xy-plane



(c) curvature clustering



(d) curvature clustering, xy-plane

embedding is to encode the similarity between nodes as distance, at least within a node's set of neighbors. In this section, we present two similar methods for constructing such a function and implement our augmentation of these in order to compare Laplacian embeddings of related data.

We begin with the Procrustes problem, named for a particularly gruesome Greek myth whose details are beyond the scope of this thesis. The Procrustes problem seeks to minimize the Euclidean distance between one set of points and the rigid rotation of another set. Let X and Y be matrices representing data sets, with data points as rows and zero-mean columns. Then the Procrustes problem [61] is to solve for an orthogonal matrix R_P such that

$$R_P = \underset{M}{\operatorname{argmin}} \|XM - Y\|_F \text{ for } M^\top M = I,$$

where $\|\cdot\|_F$ is the Frobenius matrix norm. In other words, the problem seeks the rotation which minimizes the sum of square distances between rows of X and corresponding rows of Y. If X and Y happened to be orthogonal matrices already, then this problem is solved by $R_P = X^\top Y = X^{-1}Y$. Thus, we can equivalently solve for the closest orthogonal approximation R_o to $X^\top Y$:

$$R_o = \underset{M}{\operatorname{argmin}} \|M - X^\top Y\|_F \text{ for } M^\top M = I.$$

The minimizer R_o can be calculated using the single value decomposition (SVD) of the product $X^\top Y = U\Sigma V^\top$, yielding the transformation $R_o = UV^T$. By the properties of the SVD, $UV^\top$ is orthogonal and thus the closed form solution to the Procrustes problem. Note that as an orthogonal matrix, R_P does not affect the distance or angles between points. This strategy alone not well suited for situations in which the ratio of diameters of the data sets is much larger than 1.

The Procrustes problem provides, if only preliminarily, a transformation between two LE embeddings if we expect the shapes of the embeddings to be similar in some way. Let $\Phi_\alpha(X) \in \mathbb{R}^{n\times d}$ and $\Phi_\beta(X) \in \mathbb{R}^{n\times d}$ be two such embeddings for a common data set X. Then we can form a rotation $R_{\alpha\beta} \in \mathbb{R}^{d\times d}$ which sends the α-embedding into the same space as the β-embedding: if $\Phi_\alpha(X)^\top \Phi_\beta(X) = U\Sigma V^\top$ is the SVD, then $R_{\alpha\beta} = UV^\top$.

We also present the method of Coifman and Hirn which resulted from their work on diffusion distances [32]. They begin with a similar construction of a graph from a data set, along with the further assumption that the data set changes in some way, parametrized by some parameter space $\alpha \in \mathcal{I}$. An illustrating example here would be to imagine the frames of a video; the images form pixel data sets at each time, and we expect some level of continuity of the pixel similarity over a small amount of time. Their consideration is based on a model of diffusion on the graph in which the amount of diffusion from node i to node j is proportional to the normalized adjacency $\frac{w(i,j)}{\sqrt{d(i)d(j)}}$. Then, the *diffusion distance* is defined as the L^2 norm of the difference between the diffusion at node i and that of node j, at a particular parameter value α. This distance turns out to be identical to the Euclidean distance in the LE embedding space, as the Laplacian operator L and the normalized adjacency operator $A_N = Id - L$ share the same eigenfunctions. The authors of [29] generalize their construction to a possible infinite graph, but we reduce this to the finite case. The operator which maps embedded points to β-space from the α-space $\mathcal{O}_{\alpha\beta}$ is defined as

$$\mathcal{O}_{\alpha\beta}(x_i) = \sum_{k=1}^{n} \langle \Phi^{(\alpha)}(x_i), \phi_k^{(\beta)} \rangle \phi_k^{(\beta)}.$$

As a matrix, this operator takes the much simpler form of

$$\mathcal{O}_{\alpha\beta} = (\Phi^{(\alpha)})^{\top}\Phi^{(\beta)},$$

where $\Phi^{(\alpha)}$ and $\Phi^{(\beta)}$ are matrices of eigenvectors for the α and β-embeddings, respectively.

An important note is that in the case of a finite graph, the operator which preserves diffusion distance $\mathcal{O}_{\alpha\beta}$ and the operator which solves the Procrustes problem $R_{\alpha\beta}$ are identical when $\Phi^{(\alpha)}$ and $\Phi^{(\beta)}$ each contain the full set of n eigenfunctions. To see this, we recall that the coordinates of the Laplacian embedding $\Phi_\alpha(X)$ for a data set X are given as the rows of Φ_α. If that matrix contains the full set of orthonormal eigenfunctions, then it is an orthogonal matrix, at which point matrix algebra provides a simple map between embedding spaces: $\Phi^{(\alpha)}R_{\alpha\beta} = \Phi^{(\alpha)}(\Phi^{(\alpha)})^{\top}\Phi^{(\beta)} = \Phi^{(\beta)}$. In this way, the Procrustes operator provides a *low-rank* approximate of the diffusion map operator in the case where we only collect $d \ll n$ eigenvectors.

4.2.2 LE Mappings via Clustered Rotation

Next, we deal with the situation in which two data sets $Y \subset \mathbb{R}^p$ and $Z \subset \mathbb{R}^q$ have different sizes, $|Y| = n$ and $|Z| = m$ with $m \leq n$. Assume that there is some known correspondence between a subset $X \subseteq Y$ and Z. We are then concerned with a kind of interpolation problem, where we can make an inference as to the relationship between the points in $Y - X$ and the points of Z.

Start with LE embeddings of the two sets X and Z, denoted as $\Phi_\alpha(X) \subseteq \mathbb{R}^d$ and $\Phi_\beta(Z) \subseteq \mathbb{R}^d$. Let $f : X \to Z$ be the correspondence map between sets X and Z. We wish to extend this to a function with all of Y as codomain, that is, into the ambient space of Z, $\mathbb{R}^q$.

Lift f to $F : \Phi_\alpha(X) \to \Phi_\beta(Z)$ by $F(\phi_\alpha(x)) = \phi_\beta(f(x))$. In this section and Section 4.2.3, we will detail two different strategies for extending the map f.

In the first method, we partition the embedding in order to define the mapping on different segments of the data set. We value the local geometry of the embedding in this situation much more than its global shape. As such, we can separate the embedding into pieces provided that the pieces have geometric relevance and represent important features in the data set. As described in Section 3.5, we build the Nyström extension of Φ_α on Y as $\widetilde{\Phi_\alpha} : Y \to \mathbb{R}^d \supseteq \Phi_\alpha(X)$.

Cluster the set $\Phi_\alpha(X)$ into P clusters using some clustering map $\mathcal{C} : X \to \{1, \ldots, P\}$. This map induces an equivalent clustering of Z by $\mathcal{C}_f(z) = \mathcal{C}(f^{-1}(z))$. We have explored this method using k-means clustering, hierarchical clustering approaches, as well as the curvature-based method described in section 4.1. The only requirement is that the resultant clusters are mutually disjoint. For each cluster pair $C_j = \mathcal{C}^{-1}(j) \subseteq X$ and $Z_j = \mathcal{C}_f^{-1}(j) \subseteq Z$, determine the Procrustes rotation $R_j : \mathbb{R}^d \to \mathbb{R}^d$ which minimizes the distance between $R_j(\Phi_\alpha(C_j))$ and $\Phi_\beta(Z_j)$ in the β-embedding space. This, in turn, defines $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}^d$ as the piecewise function where $\mathcal{R}(C_j) = R_j(C_j)$. According to Section 3.6, define the approximate pre-image of the β-embedding $\Phi_\beta^{-1} : \mathbb{R}^d \to \mathbb{R}^\beta$. The composition of these functions then define $f_\mathcal{C} : Y \to \mathbb{R}^q$ the *clustered* extension of f:

$$f_\mathcal{C} = \Phi_\beta^{-1} \circ \mathcal{R} \circ \widetilde{\Phi_\alpha}.$$

We designed this method, in which we partition points in the embedding space, in order to improve upon the learned correspondence between embedding spaces. This permits rotations and translations tailored to a particular region of the embedded data, rather than optimizing a global transformation. To our knowledge, this approach has not previously reported. Representative

improvements are shown in the relevant sections of Chapters 6 and 7.

4.2.3 LE Mappings via Graph Matching

Here, we present an alternative method for extending the correspondence map $f : X \to Z$ to all of Y. Instead, we will embed Y directly in the β-embedding without first passing through the α-embedding.

The perspective here is that we want to pass $f : X \to Z$ to a map between the LE embedding spaces of X and Z, extend this induced map, and then take the preimage from the embedding space of Z. Our approach bears some resemblance to methods in the problem of *graph matching* [69, 93]. In brief, the graph matching problem is usually viewed in terms of the *quadratic assignment problem*: given two adjacency matrices A,$B \in \mathbb{R}^{n\times n}$, find the permutation matrix $P \in \{0,1\}^{n\times n}$ which minimizes $tr(APB^\top P^\top)$[119]. Our problem then looks like a graph matching problem between a graph G_Y with nodes Y and a graph $G_{\mathcal{Z}}$ which contains Z. In our case, we can avoid this general NP-hard problem due to the known correspondence $f : X \to Z$.

In particular, the *functional correspondence* technique of Maks Ovsjanikov et al. [93] addresses this by seeking a map between the set of functionals on each graph. Within this framework, a node x_i on a graph G is identified with the delta function $\delta_i \in L^2(G)$. These delta functions are then expanded into the basis of Laplacian eigenvectors of G, and the coefficients are exactly the coordinates of the LE embedding $\phi(x_i)$ by Definition 3.3.6. Solving for a functional correspondence is then solved as an optimization problem which respects the geometry of the $\Phi_\alpha(X)$ and $\Phi_\beta(Z)$ [69]. Once a functional correspondence between the two dual spaces,

the authors of [93] suggest using the *iterative closest point* (ICP) algorithm to learn a pointwise correspondence between nodes of G_Y and $G_{\mathcal{Z}}$ and hence, a graph matching.

For our approach to extending the correspondence $f : X \to Z$, we again start with the function $F(\phi_\alpha(x)) = \phi_\beta(f(x))$. Recall the Nyström extension defined in Section 3.5:

$$\widetilde{\Phi}(y) = \sum_{\ell=1}^{d} \frac{1}{1-\lambda_\ell} \sum_{x_j \in \mathcal{N}_y} \tilde{w}(y, x_j) \frac{\phi_\ell(j)}{\sqrt{\tilde{d}(y) d(x_j)}}.$$

If we replace x_j with its image under f, we obtain a map from Y into the embedding space of Z that respects the geometry of both embedding spaces $\Phi_\alpha(X)$ and $\Phi_\beta(Z)$:

$$\widetilde{\Phi_\beta}(y) = \sum_{\ell=1}^{d} \frac{1}{1-\lambda_\ell} \sum_{x_j \in \mathcal{N}_y} \tilde{w}(y, x_j) \frac{\phi_\ell^{(\beta)}(f(x_j))}{\sqrt{\tilde{d}(y) d(x_j)}}.$$

That is, in words, the points in X are matched to their corresponding points in Z and embedded into the β-embedding. Then, the out-of-sample extension process is performed on the *matched* embedded points $F \circ \Phi_\alpha(X)$. From here, we can again use the approximate pre-image of the β-embedding on the image of $\widetilde{\Phi_\beta}$. The result of composing these operations is the *matched* extension $f_{\mathcal{M}} : Y \to \mathbb{R}^q$ of f.

$$f_{\mathcal{M}} = \Phi_\beta^{-1} \circ \widetilde{\Phi_\beta}.$$

Also, this matched extension can create a graph matching in the case where X and Z are both subsets of the nodes of two graphs.

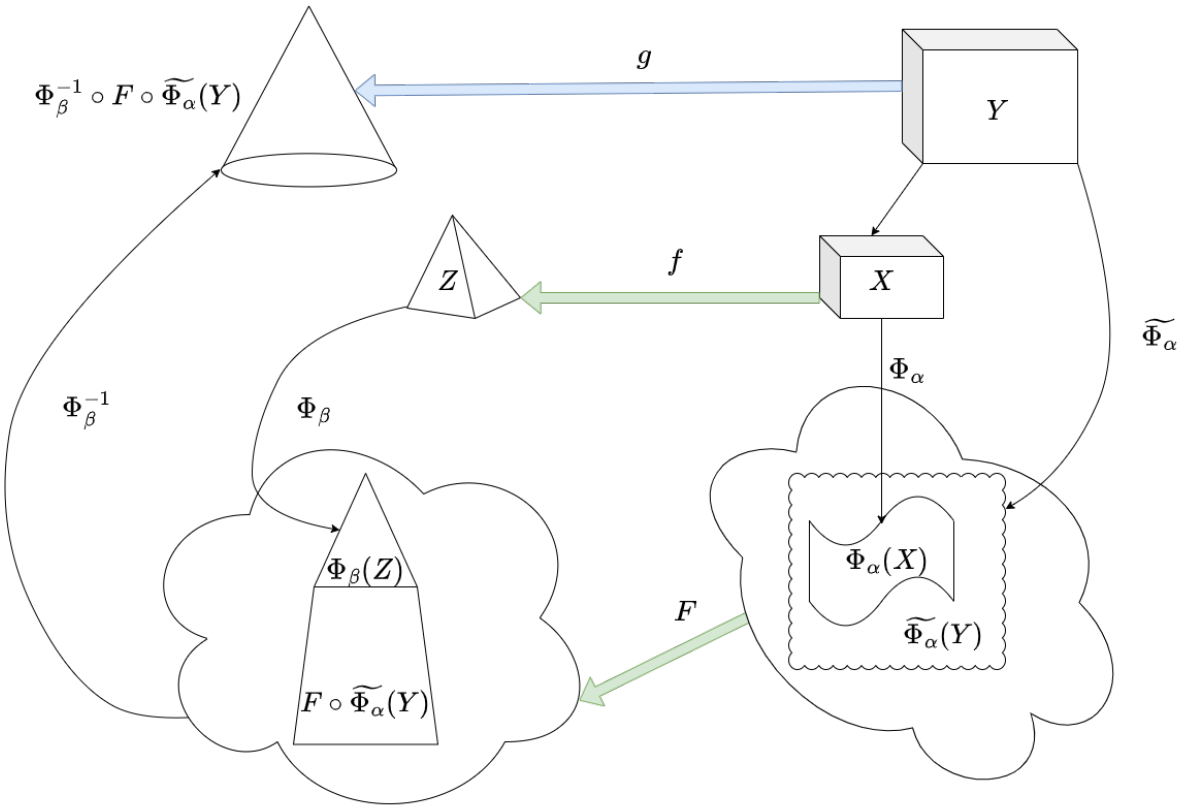
4.3 Graph Laplacian Data Fusion

Throughout this chapter, we have alluded to a particular conceptualization of a data set expressed in two modalities, α and β. For a data set Θ, we take "expressing data in two modalities" to mean the assumption that each data point has coordinates which are sampled from compact Riemannian manifolds without boundary $\mathcal{A}$ and $\mathcal{B}$. Such a *manifold learning* assumption is supported in literature describing LE and similar methods [11, 32, 89, 126]. We let $X \subseteq \mathcal{A}$ and $Z \subseteq \mathcal{B}$ represent the modality coordinates of the data set Θ. There is a natural one-to-one correspondence between coordinates of the same point in Θ, which we call $f : X \to Z$. Further, we say that these manifolds are embedded in D_α and D_β-dimensional Euclidean space, respectively. That is, $X \subseteq \mathcal{A} \subseteq \mathbb{R}^{D_\alpha}$ and $Z \subseteq \mathcal{B} \subseteq \mathbb{R}^{D_\beta}$. Under this data fusion model, the correspondence f can be extended to a function $g : \Omega_\alpha \to \Omega_\beta$ between open sets Ω_α and Ω_β which contain the manifolds $\mathcal{A}$ and $\mathcal{B}$, respectively.

A Riemannian manifold without boundary is sufficient structure for a distance metric and the *Laplace-Beltrami operator*, the Riemannian manifold generalization of the Laplacian differential operator of vector calculus. Belkin and Niyogi [11] show that under sufficient sampling conditions, that the eigenvectors and eigenvalues of graph Laplacians formed by sampling on such a manifold converge in probability to the eigenfunctions and eigenvalues, respectively, of the Laplace-Beltrami operator on that manifold. In light of this, eigenvectors of the graph Laplacian under the data fusion model are reasonable approximates to smooth functions on the underlying manifold, especially without *a priori* knowledge of such a manifold. Thus, passing the correspondence f through the LE embeddings $\Phi_\alpha(X)$ and $\Phi_\beta(Z)$, along with an inverse to the LE embedding, provides an approximation to a smooth map between Ω_α and Ω_β. The two functions

$f_{\mathcal{C}}$ and $f_{\mathcal{M}}$, the clustered and matched extensions of f, are then two ways of approximating the function g. In both of these constructions, the data points are *fused* in the LE embedding, which allows for linking between the two modalities encoded by Ω_α and Ω_β.

Figure 4.2: *A diagram illustrating the graph Laplacian data fusion model. The map $f : X \to Z$ is extended to $g : \Omega_\alpha \to \Omega_\beta$ through Laplacian embeddings Φ_α and Φ_β, the out-of-sample extension $\widetilde{\Phi_\alpha}$, and the preimage Φ_β^{-1}. This allows for Y to transfer modalities from α to β.*



The data fusion model is then the primary setting for the remaining chapters. We will show that this framework is general enough to be applicable to many supervised learning tasks in which a notion of similarity between points in a data set is of particular importance. In the subsequent chapter, we explore an implementation the data fusion model for outlier detection. Given data sets X and Z as above, as well as the correspondence f and its extension, for instance

$f_{\mathcal{C}}$, $\|f(x_i) - f_{\mathcal{C}}(x_i)\|_{\Omega_\beta}$ measures the extent to which contextual information in modality α can be transferred to modality β for some point $x_i \in X$. Points for which this value is high serve the highlight the idiosyncrasies of modality α with respect to β, furthering understanding of the link between the two.

The final two chapters of this thesis are concerned with the application of the data fusion model in a multimodal superresolution problem.

4.4 Superresolution using Laplacian Eigenmaps

Here we define the multimodal superresolution (MMSR) problem which we aim to solve, with our attempts detailed in Chapters 6 and 7. Suppose we have two images of the same subject but in two different modalities, modality α and modality β. Suppose also that there is some quantitative relationship between the modalities. Then, given a high resolution image in modality α and a low resolution image in modality β, both of the same subject, we want to find an approximate high resolution image in modality β.

For an example of such a relationship between modalities, consider a digital color image and a grayscale version of that same image. With immense loss of generality, choosing to ignore most of the human biology, psychology, physics, and engineering involved in human perception of contrast and color, we can use the National Television System Committee (NTSC) standard conversion from RGB color to *luminance*, or black-and-white intensity, as the formula relating color and grayscale: $y = 0.2989R + 0.5870G + 0.1140B$. Then the more interesting of the two versions of the problem in this case is recovery of a high resolution color image given a low resolution color and high resolution grayscale image. One would hope that the additional color

information from the low resolution image helps to offset the fact that the map from color to gray is not injective.

One can think of this problem as having two distinct facets: the more traditional, single image superresolution problem of upscaling a low resolution image, and a supervised learning problem of discovering characteristics of modality β using limited information about the relationship between α and β. We approach this problem by framing it within the context of Laplacian eigenmap embeddings and the data fusion model. Individual pixels in an image can be assigned vectors, from which we can construct Laplacian embeddings. With sufficient choices of pixel vector and embedding hyperparameters, we can encode contextual modality information within the geometry of the embedding space. In this way, if we have a map between embedding spaces which respects their geometries, then we can essentially transfer a pixel in modality α to one in β. Thus, we can upscale a low resolution β image to at least the resolution of the equivalent α image. The rest of this section details the method by which we solve this problem in two different cases of modality pairs.

4.4.1 Constructing Pixel Vectors

In Section 3.4, we introduced the general method for obtaining a graph from the pixels in an image. Here, we elaborate on that construction for the MMSR problem: given a low resolution m_1 by m_2 image $\mathcal{I}_\beta^{\mathcal{L}}$ and a high resolution n_1 by n_2 image $\mathcal{I}_\alpha^{\mathcal{H}}$, obtain a high resolution n_1 by n_2 approximate image $\widehat{\mathcal{I}}_\alpha^{\mathcal{H}}$ in the β-modality. For n_1 by n_2 grayscale image $\mathcal{I}$, we will use $vec(\mathcal{I}) \in \mathbb{R}^n$ to denote the column vector of pixel values of $\mathcal{I}$, where $n = n_1 \cdot n_2$. From these two images $\mathcal{I}_\beta^{\mathcal{L}}$ and $\mathcal{I}_\alpha^{\mathcal{H}}$, we will construct matrices Z and Y, respectively. The rows of

these matrices will then be the data vectors for pixels in each image, with notation for the data matrices in accordance to Sections 4.2.2 and 4.2.3. At this point, we describe the formation of the data vectors are for each pixel. In general, we describe this process on images for which only one number is assigned to each pixel, such as grayscale images. For color or other multichannel images, the procedure is performed on each channel independently and the results are recombined where appropriate to create pixel vectors.

4.4.1.1 Spatial Image Data

First, we form the data matrix Z whose rows will be the data vectors for the pixels in the low resolution image $\mathcal{I}_\beta^{\mathcal{L}}$. Start with the vectorized image $vec(\mathcal{I}_\beta^{\mathcal{L}}) \in \mathbb{R}^m$, where $m = m_1 \cdot m_2$. Specifically, we follow a *column-major* order in vectorization so that the first m_1 entries in $vec(\mathcal{I}_\beta^{\mathcal{L}})$ consist of the values of the first column on the left of the image $\mathcal{I}_\beta^{\mathcal{L}}$. Next, we adjoin the spatial location information of each pixel in the following way.

Let $loc_x(\mathcal{I}_\beta^{\mathcal{L}})$ be the m_1 by m_2 matrix with values in $[0, 1]$ such that $loc_x(\mathcal{I}_\beta^{\mathcal{L}})(i, j) = \frac{j}{m_2}$. Similarly, let $loc_y(\mathcal{I}_\beta^{\mathcal{L}})$ be the m_1 by m_2 matrix with values in $[0, 1]$ such that $loc_y(\mathcal{I}_\beta^{\mathcal{L}})(i, j) = \frac{i}{m_1}$. In other words, these two matrices give the horizontal and vertical location, respectively, of each pixel in $\mathcal{I}_\beta^{\mathcal{L}}$. In this way, we form the n by 2 matrix $Spat(\mathcal{I}_\beta^{\mathcal{L}}) := [vec(loc_x(\mathcal{I}_\beta^{\mathcal{L}})), vec(loc_y(\mathcal{I}_\beta^{\mathcal{L}}))]$.

The data matrix Z for $\mathcal{I}_\beta^{\mathcal{L}}$ is then $Z = [vec(\mathcal{I}_\beta^{\mathcal{L}}), Spat(\mathcal{I}_\beta^{\mathcal{L}})]$. Let Y be the data matrix whose rows are the data vectors for the high resolution image $\mathcal{I}_\alpha^{\mathcal{H}}$. Then Y is formed similarly as Z, where $Y = [vec(\mathcal{I}_\alpha^{\mathcal{H}}), Spat(\mathcal{I}_\alpha^{\mathcal{H}})]$.

4.4.1.2 Joint Modality

Here, we detail yet another addition we can make to the data vectors for the images in the superresolution problem. This involves including the information from the low resolution modality α image in the data matrix Y itself. Li [77] demonstrates the ability for so-called *fused* images to encode detail information from both source images. Dong [45] and Lu [83] show the use of a fused image in a superresolution problem under different sparse encoding schemes. Inspired by those works, we propose incorporating a joint α and β modality image into the data matrix Y.

In this situation, the data matrix Z for the low resolution image $\mathcal{I}_\beta^\mathcal{L}$ remains unchanged from the construction in the preceding section. The method of Lu et al. [83] involves a very similar superresolution problem to ours, except from the perspective of teasing out β image features from a composite high resolution image containing information from both modalities. In their method, they begin with the creation of a high resolution approximate β image via some interpolation technique. Using the Fourier wavelet-based method of Li et al. [77], the high resolution approximate is *fused* with the high resolution modality α image. This high resolution fused image is a mix of the two modalities, and they isolate the β image using a deblurring technique of Dong and collaborators [45]. This is done by constructing a *local linear embedding* (LLE) using patches of the high resolution fused image, with patches being analogous to pixels in our construction. LLE is a kernel method similar to LE, expanding on eigenvectors a slightly different matrix function of a graph adjacency. Finding the high resolution β image is then done with an optimization whose loss function includes a term enforcing that the solution share the same LLE structure as the high resolution fused image. We adapt their joint image idea by first upscaling $\mathcal{I}_\beta^\mathcal{L}$ using some

upscaling function $Big : \mathbb{R}^{m_1 \times m_2} \to \mathbb{R}^{n_1 \times n_2}$. We choose Big to be a bicubic interpolation in most cases, as it is an effective upscaling method which is computationally inexpensive relative to other calculations involved in the algorithm.

The upscaled image $Big(\mathcal{I}^{\mathcal{L}}_{\beta})$ is then the same size as $\mathcal{I}^{\mathcal{H}}_{\alpha}$ with pixels corresponding to roughly the same spatial location. We can then consider the joint image $\mathcal{I}^{\mathcal{H}}_{\alpha\beta} = \mathcal{I}^{\mathcal{H}}_{\alpha} \oplus Big(\mathcal{I}^{\mathcal{L}}_{\beta}) \in \mathbb{R}^{n_1 \times n_2 \times 2}$ of high resolution as one multichannel image. The quality of the alignment between $Big(\mathcal{I}^{\mathcal{L}}_{\beta})$ and $\mathcal{I}^{\mathcal{H}}_{\alpha}$ certainly has the potential to reduce overall results, but we emphasize that the upscaled image primarily functions within the joint image to give coarse β information, as indeed we lose precise spatial already during through the upscaling. In the example of high resolution color recovery, this can help disambiguate the hue of two pixels which produce similar grayscale values. At this point, we add the spatial information of $\mathcal{I}^{\mathcal{H}}_{\alpha}$ to create the data matrix $Y = [vec(\mathcal{I}^{\mathcal{H}}_{\alpha}), vec(Big(\mathcal{I}^{\mathcal{L}}_{\beta})), Spat(\mathcal{I}^{\mathcal{H}}_{\alpha})]$.

Now that we have Z and Y, we want a representative set X to form a bijection with Z. That is, we need to establish a set of vectors representing pixels in the α modality which initialize the connection between α and β. Let $Small : \mathbb{R}^{n_1 \times n_2} \to \mathbb{R}^{m_1 \times m_2}$ be a downsizing operation. This can be something like regular sampling on the grid of $n_1 \times n_2$ points or an interpolation down to the desired size. We choose a bilinear interpolation, and in the vast majority of our examples where $\frac{n_1}{m_1} = \frac{n_2}{m_2}$, this amounts to averaging over a small square patch of the high resolution image. We use our downsizing operation on the joint image $\mathcal{I}^{\mathcal{H}}_{\alpha\beta}$ to obtain $\mathcal{I}^{\mathcal{L}}_{\alpha\beta} = Small(\mathcal{I}^{\mathcal{H}}_{\alpha} \oplus Big(\mathcal{I}^{\mathcal{L}}_{\beta})) = Small(\mathcal{I}^{\mathcal{H}}_{\alpha}) \oplus Small(Big(\mathcal{I}^{\mathcal{L}}_{\beta}))$. To clarify, we downsize both channels of the joint image separately, then join the downsized versions back together. An important point here is the whole of the relationship between modalities α and β, as it pertains to our method, rests on the assumption that pixels at the same spatial location between $\mathcal{I}^{\mathcal{L}}_{\alpha\beta}$ and $\mathcal{I}^{\mathcal{L}}_{\beta}$ point to the

same object in an image. In particular, we want to make sure that the $Small$ operator allows for this and doesn't introduce shifts or spatial distortion in the image. This is akin to making sure that our training data has correct ground truth labels.

Finally, we vectorize $\mathcal{I}^{\mathcal{L}}_{\alpha\beta}$ and add the spatial information from $Small(\mathcal{I}^{\mathcal{H}}_{\alpha})$ to create the matrix $X = [vec(Small(\mathcal{I}^{\mathcal{H}}_{\alpha})), vec(Small(Big(\mathcal{I}^{\mathcal{L}}_{\beta}))), Spat(Small(\mathcal{I}^{\mathcal{H}}_{\alpha}))]$. Note that in Sections 4.2.2 and 4.2.3, we present almost this exact setup, a key difference being that the condition $X \subseteq Y$ is no longer required. This is equivalent to $Small$ being a sampling operator. Our goal is to create two Laplacian embeddings which connect the modalities and X and Z serve that purpose. As the vectors in X come from downsized versions of the vectors in Y, the Euclidean distance, our measure of similarity for pixel vectors, between points in Y and Z is just as valid as between two points in Y. Recall that $f : X \to Z$ is how we associate points in the low resolution version of both modalities. Then from here, we can follow either the clustered extension $f_{\mathcal{C}}$ of f along with some clustering method or the matched extension $f_{\mathcal{M}}$ of f.

The matched extension $f_{\mathcal{M}}$ depends on the quality of the Nyström extension that can be created using the points in X. For example, if we downsize by only taking some $m_1 \times m_2$ patch of the image, we may not have enough information about modality β for the Nyström extension into the β space to give good results. Clustering highly weights the relationships among points of Y, and so we may expect this to be a superior method if our modality α image avails itself to easy segmentation. We thus suspect that clustering has the potential to work better for multimodal superresolution in irregular downsizing situations than matching, as the location of points in the embedded β space depends on relationships within Y as much as, if not more so, the map f between X and Z. Validating this conjecture would take additional study and provides an interesting future research question.

Chapter 5: Early Work – Cell Line Data Fusion

5.1 Outline

In this chapter, we present work published in 2017 by Jeremiah Emidih and Wojciech Czaja [35] on using an early version of our data fusion model, presented in Section 4.3, in predicting drug response. We are assisted by genomic research conducted by Yves Pommier, Vinodh Rajapakse, and the LMP at the NCI-Developmental Therapeutics Branch in curation of the data set, providing necessary genomic background, and building the prediction function. Alexander Cloninger and Timothy Doster are also specially acknowledged for making the data fusion model viable through their creation of a stable preimage algorithm for Laplacian eigenmap (LE) embeddings. I wrote the text, designed and conducted the computational experiments, and found the genes which were used to predict drug response.

This early work allowed me to explore the capabilities of the data fusion model when applied to highly nontrivial, nonlinear data. Here, the data points are genes, given as vectors of gene expression across certain cancer cell lines. These genes are expressed in two modalities; in terms of their gene expression in cancer cell lines which are either resistant or sensitive to a particular drug used in some cancer treatments.

5.2 Heterogeneous Cancer Cell Line Data Fusion for Identifying Novel Response Determinants in Precision Medicine

5.2.1 Background

Genomic analysis is required for precise medicine to find the right drug for the right patient. In order to do this accurately, one must integrate terabytes of data over dozens of databases, filled with data of varying quality and content.

Cancer cell line databases have collected data on over 1,300 cell lines representing the broad spectrum of human cancers. Specifically, there is extensive analysis of genomic and drug response data on the cell lines in the National Cancer Institute (NCI-60) database and the Broad Institute Cancer Therapeutics Response Portal (CTRP) database [1, 100].

Topoisomerase inhibitors are among the most effective and widely used anticancer drugs, and in spite of known molecular pathways for DNA repair, it is not currently possible to predict how cells respond to these DNA-targeted drugs [97]. Using standard statistical tools, the Pommier laboratory (LMP) of the NCI Developmental Therapeutics Branch (NCI-DTB) discovered new factors to determine topoisomerase inhibitor response from databases based on the NCI-60 [1, 111, 127]. These include previously unsuspected response determinants like the putative helicase SLFN11 [115, 127].

The NCI-DTB uses CellMiner web-based suites to build prediction functions from these response determinants based on the elastic net regression algorithm, which was applied previously for drug response prediction for other databases [57, 84, 101]. Elastic net is a linear regression model which adds a quadratic penalty to the loss function from LASSO regression. An issue

with this method though is that highly correlated response determinants, such as genes that are co-expressed, can be swapped to obtain different predictive models which are very similar in predictive performance.

Thus, the goal of this work is to find novel response determinants to topoisomerase inhibitors in a method independent of features selected by elastic net. In particular, we shall use a nonlinear feature selection model to avoid the aforementioned problem with a variety of equivalent predictive functions. The features to be selected are among the cell line database information which gives gene expression values indicating log2 probe intensity using the Affymetrix HG U133 Plus 2.0 Microarray platform [100]. The main databases considered, NCI-60 and CTRP, have drug response and gene expression information for the cell lines, so the feature selection method must appropriately integrate the two types of data. Our approach will attempt this by examining the expression of genes on cell lines that are particularly sensitive or resistant to the topoisomerase inhibitor Topotecan.

5.2.2 Method

The goal is to utilize a nonlinear feature selection and data integration, or *data fusion*, to find predictive response determinants. To do this, we reformulate the problem from a mathematical perspective: given a data set which is expressed in two different modalities, find a similarity metric and elements of the set whose expressions differ beyond an appropriate threshold.

We start with a data set X of size $|X| = m$ that is expressed in two modalities a and b, which we represent with data matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{m\times p}$, respectively. For a data point $x \in X$, let $A(x)$ denote its expression in modality a and similarly for $B(x)$. In general, n is

not equal to p, which makes naive comparison of n-dimensional $A(x)$ and p-dimensional $B(x)$ difficult.

Our method uses three main steps: embed the data matrices onto feature manifolds, create a joint representational feature space, and use a stable preimage mapping to facilitate comparison.

5.2.2.1 Manifold learning data representation for biological process-based feature selection

Laplacian eigenmaps (LE) is a nonlinear topology-preserving transformation technique that is designed to embed data onto a manifold based on local neighborhoods of data points [10]. We use LE to transform the data set X into two feature spaces Γ_a and Γ_b which capture intrinsic information of X based on the modalities a and b. The mappings are given by $\phi_a : X \to \Gamma_a$ and $\phi_b : X \to \Gamma_b$, where $m_a = dim(\Gamma_a)$ and $m_b = dim(\Gamma_b)$ are chosen to be greater than n and p, respectively, to avoid unnecessary loss of information due to dimension reduction.

5.2.2.2 Data fusion through joint feature space representation

The embedded points in Γ_a are now of the form $\phi_a(x) = (\phi_a^i(x))_{i=1}^{m_a}$, with a similar form for points in Γ_b. At this step, we can embed the points in Γ_a into Γ_b using the Coifman-Hirn rotation operator $\mathcal{O}_{ab}$. For a data point $x \in X$, $\mathcal{O}_{ab}(x)$ is given by

$$\mathcal{O}_{ab}(x) = \left(\sum_{i=1}^{m_a} x_i \langle \phi_a^i, \phi_b^j \rangle \right)_{j=1}^{m_b},$$

where $x_i = \phi_a^i(x)$. Now, each data point in X is represented by two points in Γ_b, one for each modality. This technique, matching features using a spectral rotation in the feature space, has

been used previously for heterogeneous image data [14, 29, 37, 39].

5.2.2.3 Inverse mapping for finding potential response determinants

LE is a nonlinear, continuous embedding of points in Euclidean space into a smooth manifold, so finding preimages of points is highly non-trivial, especially when the points are not in the convex hull of the training points used to form the manifold. Alexander Cloninger, Wojciech Czaja, and Timothy Doster solve this issue by developing a fast and accurate preimaging algorithm for LE which exploits the sparsity of certain matrices constructed in the LE process [31]. By

Let $\widetilde{\phi_b} : \Gamma_b \to \mathbb{R}^p$ denote this approximate preimage, and let $y = \mathcal{O}_{ab}(x)$ for a data point $x \in X$. Then $B(x)$ and $\widetilde{\phi_b}(y)$ are both data vector representations in $\mathbb{R}^p$ of the same point x which can be easily compared using a preferred norm. We denote the data matrix of these preimage points as $\widetilde{A} \in \mathbb{R}^{m \times p}$. An advantage of comparing points in the original space $\mathbb{R}^p$ as opposed to comparing in the representational feature space Γ_b is that qualitative meanings of points in the original space may change in the feature space.

This entire process is symmetric in modalities a and b, providing a multitude of options for comparing points.

5.2.3 Preliminary Results and Future Work

The CTRP dataset we used contains 1283 genes expressed over the 792 cell lines that have response data for Topotecan. The genes chosen are known to have broad relevance to cancer and pharmacology. They were also filtered such that the upper quartile gene expression across cell

lines was over 5 and the range of gene expressions across cell lines was over 2. This curation is for three reasons: lowly expressed genes may not affect the cell in substantive ways, genes with low variability are not reliable response determinants, and the preimage mapping uses an optimization technique which is sensitive to very small values. Let G be the set of these genes.

The cell lines were subsetted and classified by having Topotecan responses 2 standard deviations above and below the mean response, creating a set of 13 resistant cell lines and a set of 18 sensitive cell lines, respectively. We represent gene expression along resistant and sensitive cell lines as data matrices $R \in \mathbb{R}^{1283\times 13}$ and $S \in \mathbb{R}^{1283\times 18}$, respectively. Within the context of the feature selection algorithm, sensitivity and resistance to Topotecan are the modalities over which G is described.

We then obtain LE embeddings $\phi_r(G)$ and $\phi_s(G)$, where $m_r = m_s = 40$ was chosen for computational ease. Following the rest of the algorithm twice, embedding Γ_r into Γ_s and vice versa, we obtain two additional data matrices:

$$\widetilde{R} = \widetilde{\phi_r} \circ \mathcal{O}_{sr} \circ \phi_s(G) \in \mathbb{R}^{1283\times 18}$$

and

$$\widetilde{S} = \widetilde{\phi_s} \circ \mathcal{O}_{rs} \circ \phi_r(G) \in \mathbb{R}^{1283\times 13} \quad .$$

We then computed the match distances for each gene $g \in G$, given by

$$d_r(g) = \left\| \widetilde{S}(g) - R(g) \right\|_2$$

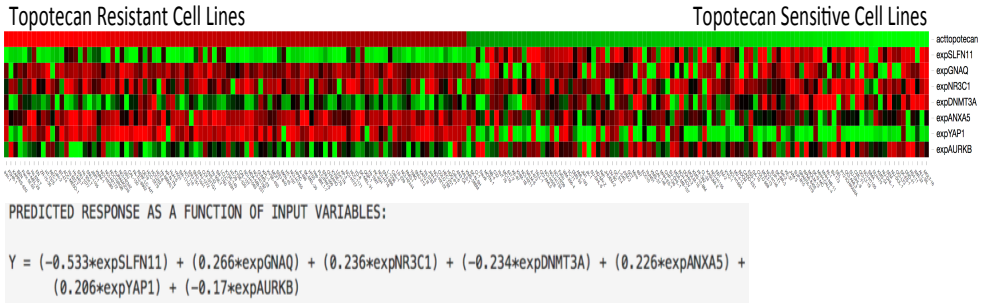


Figure 5.1: *A heatmap view of drug activity and gene expression data for 200 CTRP cell lines with the most extreme responses to Topotecan. Higher and lower values are represented by red and green, respectively. The LASSO-derived regression model augments the predictive capacity of SLFN11 expression, with a (10-fold cross-validation) predicted vs. observed drug response value Pearson's correlation of* 0.67*, versus* $r = 0.45$ *for SLFN11 alone.*

and

$$d_s(g) = \left\| \widetilde{R}(g) - S(g) \right\|_2 .$$

Let δ_r, δ_s be the mean plus 1 standard deviations of the sets $d_r(G)$ and $d_s(G)$, respectively. Potential response determinants were thus elements of the set

$$G' = \{g \in G | d_r(g) > \delta_r \text{ and } d_s(g) > \delta_s\}.$$

G' contains 38 genes and includes SLFN11, which was previously discovered to be a response determinant using statistical methods and then further validated. This provides a promising start toward finding response determinants with low ambiguity that give more accurate predictive models.

We plan on repeating this method on DNA-targeted drugs such as MK-1775 and the PARP inhibitor Olaparib. Eventually, we aim to make the entire data integration paradigm more robust by examining other operators on the feature spaces and other nonlinear embeddings based on data-dependent graphs.

Chapter 6: Multimodal Superresolution in Color Images

6.1 Outline

The goal of this chapter is to develop a method for solving a mulitimodal superresolution (MMSR) problem in small color images. We aim to recover a high resolution version of a color image using a low resolution color version and a high resolution black and white (BW) version of the same image. This will be done using a combination of the methods described in Chapter 3. We will demonstrate the usefulness of the described methods, eventually converging on a usable pipeline.

This problem is partially motivated by a paper from Baatz, Fornasier, Markowich, and Schönlieb [4] in digital inpainting. In that paper, they describe their approach in restoring 14th century Neidhart wall frescoes, rediscovered in 1979 by happenstance behind plaster walls in an unassuming apartment in Vienna. Degradation of the paint pigments over time have altered the saturation, hue, and contrast. The authors develop a method using partial differential equations to restore parts of the image with limited color and contrast information.

In order to obtain a recreation of the structure of the fresco, they need to infer grayscale information from their limited and altered color data. In our MMSR problem, we have a similar constraint for reversed modalities. However, we test our color MMSR implementation on much smaller images with a more restricted range of colors than the Neidhart fresco. This is so that we

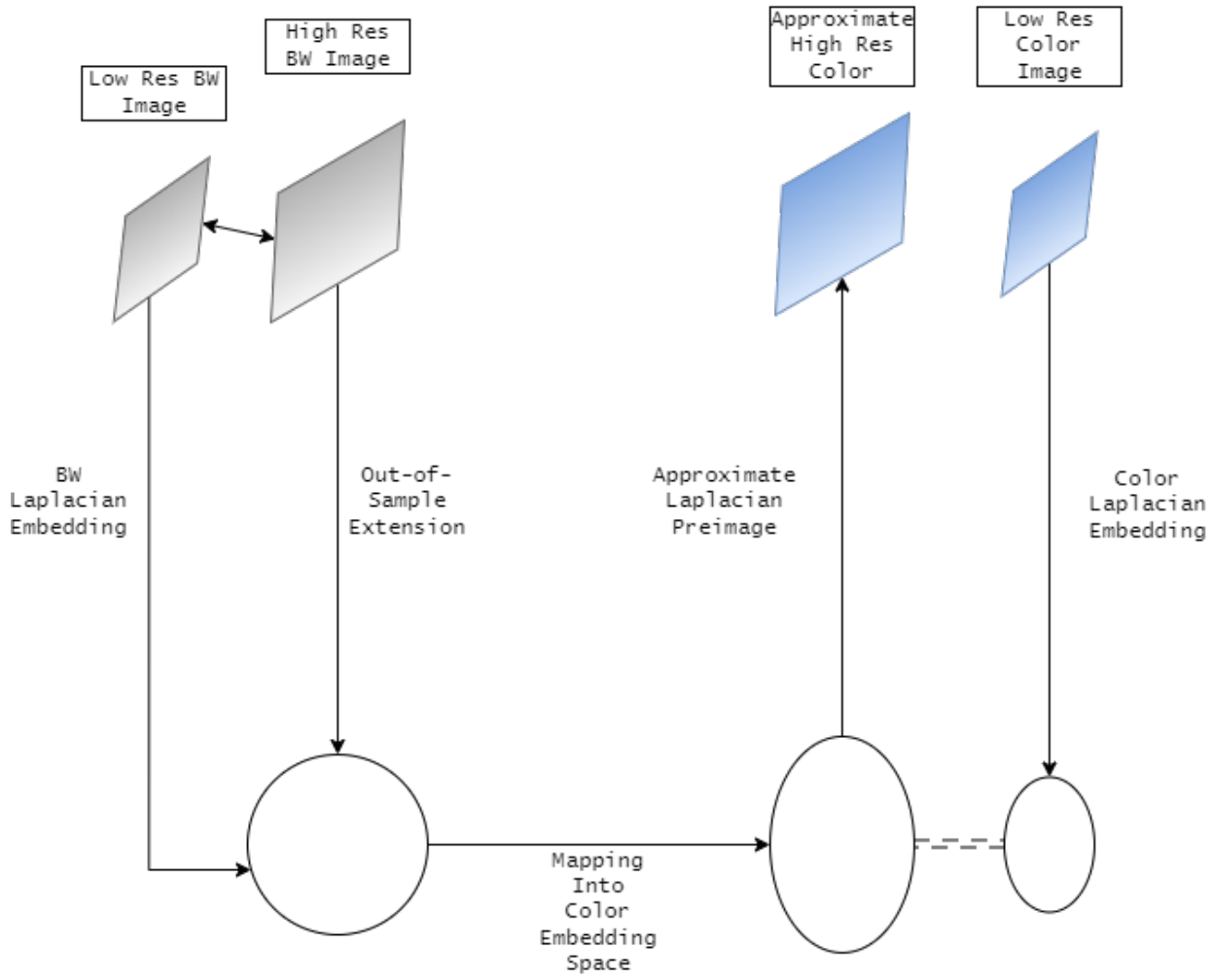
Figure 6.1: A part of the Neidhart fresco found in 1979. Over time, the pigments have degraded, making restoration by inpainting a difficult problem.
Image courtesy of the Vienna Museum

can make visual assessments of the quality of reconstructions easily, without the need for subject matter expertise.

6.2 CIFAR-10 Data and Evaluation Metrics

The data set we use here is the CIFAR-10 data set [70] which consists of many 32×32 images across 10 classes. The images were selected from a larger set, extensively curated by internet search, with keywords all chosen to represent common English words [117]. CIFAR-10 is among the most widely used natural image data sets in the fields of machine learning and computer vision. To our knowledge, the is no standard image data set constructed for the purpose of MMSR as we describe. Thus, we create a corresponding set of black-and-white (BW) images from the CIFAR-10 set. These color images are converted to BW images using the built-

Figure 6.2: A diagram of the color MMSR algorithm



in MATLAB function rgb2gray, which applies the linear transformation $0.2989R + 0.5870G + 0.1140B$ [80] to the red, green, and blue color channels.

Figure 6.3: *Example Image of a Bird. The low resolution image (b) is 16 by 16 pixels while the high resolution images (a), (c) are 32 by 32 pixels.*

(a) *High Resolution BW* **(b)** *Low Resolution Color* **(c)** *High Resolution Color*

Figure 6.4: *Example Image of a Cat. The low resolution image (b) is 16 by 16 pixels while the high resolution images (a), (c) are 32 by 32 pixels.*

(a) *High Resolution BW* **(b)** *Low Resolution Color* **(c)** *High Resolution Color*

We will demonstrate these methods using one example image each from the "Bird" and "Cat" classes, which will be referred to as the Bird and Cat images, respectively. First, we show the low resolution color image created by taking average intensities over square pixel patches, the high resolution BW image, and the original high resolution color image that we wish to recon-

struct. Low resolution images are chosen to have one-fourth the pixels as the high resolution, half the size in each spatial dimension. Then, we show the embeddings which result from these images, using the method of Sections 3.4 and 4.2.2. Specifically, we are performing the Procrustes rotation but taking the entire set of embedded points as one cluster, yielding a *global rotation*.

In addition to visual inspection, the quality of the reconstructions is measured using both peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM). The SSIM is a value which measures the similarity of image A when compared to image B considering luminance, structure, and contrast. If we take μ_A and μ_B as the means of images A and B, σ_A and σ_B as the standard deviations, and σ_{AB} as the cross-covariance between the two, then the SSIM between the two is calculated as follows:

$$SSIM(A,B) = \frac{(2\mu_A\mu_B + 0.01^2)(2\sigma_{AB} + 0.03^2)}{(\mu_A^2 + \mu_B^2 + 0.01^2)(\sigma_A^2 + \sigma_B^2 + 0.03^2)}.$$

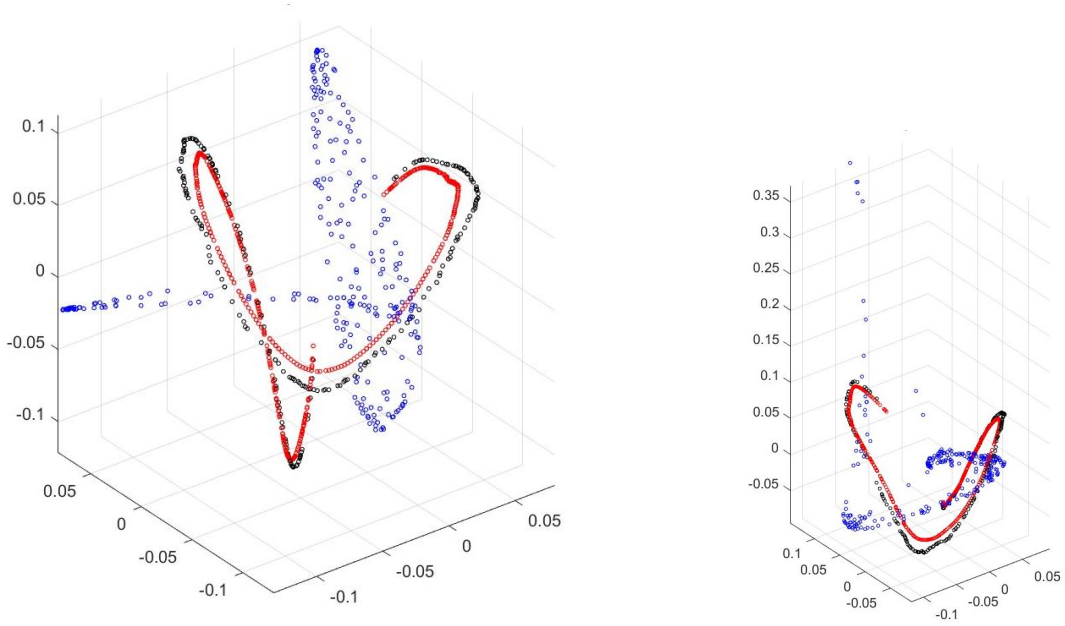
Let $M = \frac{1}{n}\|A - B\|_2^2$ be the mean squared error between images A and B. Then the PSNR of A when compared to B is $P = 10\log_{10}(\frac{1}{M})$. These two metrics are standard for image processing tasks and can be shown to be equivalent in certain instances [65].

6.3 Spatial Data in LE Embeddings of Color Images

These first embeddings in Figure 6.5 follow the construction outlined in Section 3.4 for the bird and cat images. That is, the representative vectors for each pixel use only the intensity values, not the spatial relationships of the pixels themselves. It is well known that the pixels in natural images like those found in the CIFAR-10 set are highly correlated in that they represent the same image features. As such, we expect that the embedding representations would be more suitable

for reconstruction if we form our pixel vectors incorporating spatial data using the method of Section 4.4.1.1.

Figure 6.5: *LE embeddings of BW and color Bird images, both on the same plot in (a) and LE embeddings of BW and color Cat images, both on the same plot in (b). The black and red points represent the low and high resolution BW pixels, respectively, while the blue points represent the color pixels. The low resolution color points appear as a 2-dimensional surface, while the BW points look like 1-dimensional curves.*



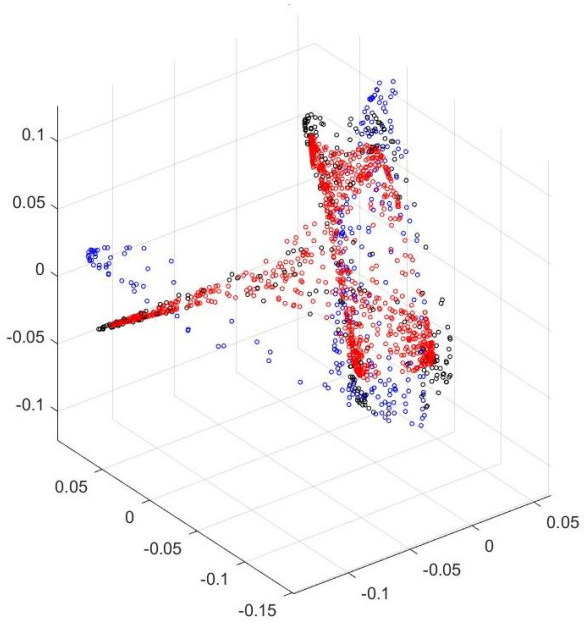
(a) *Bird Image Plots*

(b) *Cat Image Plots*

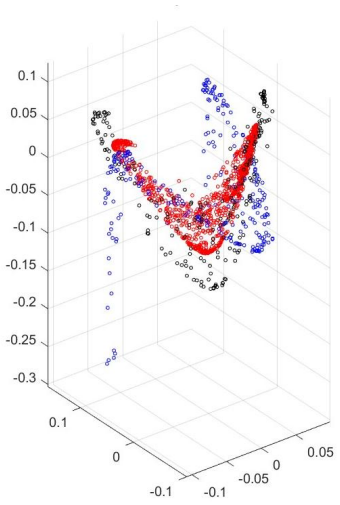
The image graphs, both low resolution BW and low resolution color, are constructed using 40 nearest neighbors, relative to the size of the image, and σ is chosen to be the 80-th percentile of all distances between neighboring pixel vectors. σ was chosen this way as an adaptation of the common decision to base σ on the empirical distribution of squared distance in the data [81]. Both LE embeddings are created using the 5 eigenvectors of each graph Laplacian corresponding to the smallest 5 nonzero eigenvalues. All depictions of these embeddings are in along the first 3 eigenvectors.

Next, we add spatial information as outlined in Section 4.4.1.1. In our example, rescale the

Figure 6.6: *LE spatial embeddings of BW and color Bird images, both on the same plot in (a) and LE spatial embeddings of BW and color Cat images, both on the same plot in (b). The black and red points represent the low and high resolution BW pixels, respectively, while the blue points represent the color pixels. When compared to Figure 6.5, the BW points no longer form 1-dimensional curves after the inclusion of spatial information, but form surfaces which more closely resemble those of the color points.*



(a) *Bird Image Plots*



(b) *Cat Image Plots*

bird image spatial values by 0.2 and the cat image spatial values by 0.1 to control the influence spatial data has on the graph construction. These values were chosen by testing a range of values between 0 and 1 and visual inspection of the connectivity of the resulting embedded points. There are a few immediate differences to take note of in Figure 6.6 once spatial information is included. For one, the shapes of the embedded points, both in- and out-of-sample, depicted in red and black, respectively, seem to be of a different dimension than before the inclusion of spatial data. On the embeddings in Figure 6.5, the BW points form curves, with pixels of extreme intensity values at the ends of the curves. We attribute this to the inherent 1-dimensional nature of the data; the LE embedding is organizing the data by grayscale value. This provides a sanity check that relevant information is conveyed in the process. However, it makes it less plausible that there would be a reasonable match between these curves and the surface-like blue embedded points of the low resolution color images.

With the inclusion of spatial data, all the embeddings appear as surfaces or point clouds in the first 3 eigenvectors, not as curves. However, the scale of the embedding, measured as the range of coordinate values achieved in each dimension, does not seem to change much after spatial data is added. Distance in the embedding space reflect similarity between pixel vectors. Thus, we attribute this stability in scale to an invariability of strength of connections between pixel vectors before and after spatial data is added. As an example, Figures 6.5b and 6.6b both feature a set of points from the color embedding which form a sparse line segment moving up and down the z-axis, respectively. This indicates the presence of some pixel vectors which are very weakly connected to the rest of the set, and the strength of the connections does not appear to change even with spatial data. Adding spatial data does change the overall shape of the embedding, seemingly in a way that benefits direct comparison of two embeddings. Indeed, we see that this

trail of points is shorter, spanning from about $z = -3$ to $z = 0$ in Figure 6.6b, than its counterpart in Figure 6.5b, which spans from below $z = -0.05$ up to $z = 0.35$.

6.4 Color Superresolution by Rotation

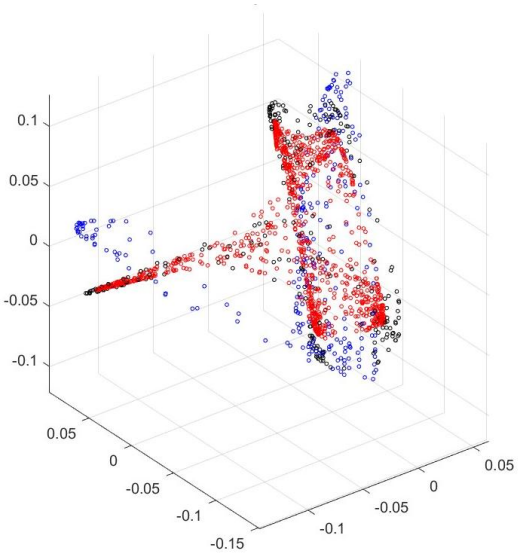
6.4.1 Global Rotation

In this section, we examine the results performing a global rotation on out-of-sample BW points, such as those indicated in red on Figures 6.6a and 6.6b. From here, the preimages are calculated for each point to recover a high resolution color image. We show the results of the preimage on the Bird and Cat examples on Figures 6.8 and 6.9. Referring to Figure 6.2, the mapping between LE embedding spaces is a global rotation, i.e. a Procrustes rotation on the entire set of embedded points, from the BW embedding into the color embedding.

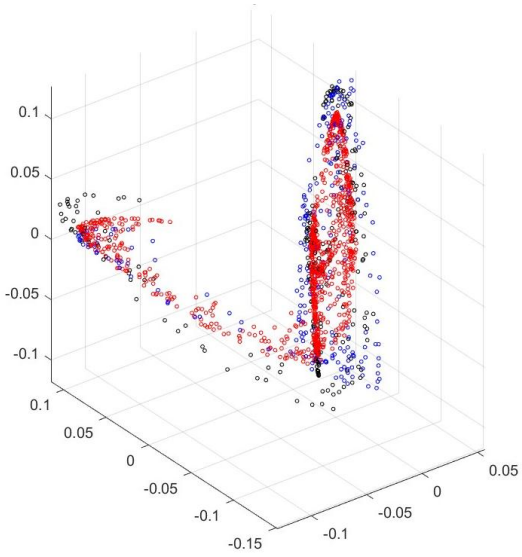
First, we examine the results of the rotation in the color embedding spaces for both Bird and Cat. In terms of the colors in Figure 6.7, each rotation is built to minimize the distance between the low resolution BW and color points. Note that the rotation takes place in the full 5-dimensional embedding space, and we are only seeing the projection of that embedding along the first 3 eigenvectors. This means that while distances and angles between points are preserved on the whole, our projection may not appear like that is the case.

The rotation result in Figure 6.6a shows an overall match in shape between the two embeddings. We can visualize that as a result of the out-of-sample extension which brings the high resolution BW pixels into the LE embedding, the high resolution BW points on the plot fit within the convex hull of the low resolution BW points. On the other hand, Figure 6.6b shows that these low resolution BW points are unable to properly match the trail of color points traveling

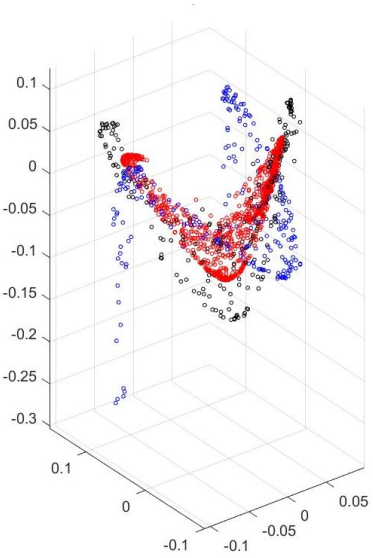
Figure 6.7: *The effect of a global rotation applied to BW LE embedding into the color embedding space for both Bird and Cat images. The black and red points represent the low and high resolution BW pixels, respectively, while the blue points represent the color pixels. For the Bird image in (b), the BW points appear to fit the shape of the color points. However, for the Bird image, there is a size discrepency between the two embeddings, noticeable in (c) as a trail of color points parallel to the z-axis. The rotation (d) of the BW embedding does not fill out the space of the color embedding.*



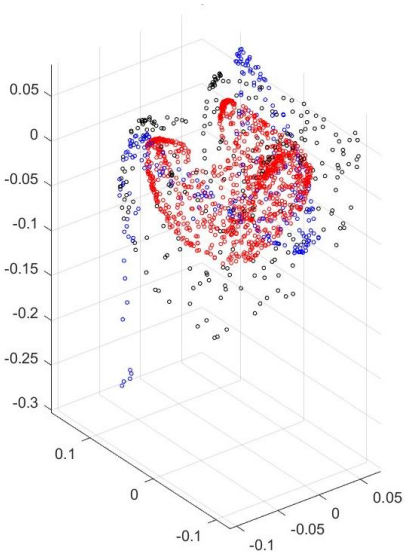
(a) *Bird image*
Before rotation



(b) *Bird image*
After rotation



(c) *Cat image*
Before rotation



(d) *Cat image*
After rotation

down the z-axis. This trail of color points roughly corresponds to pixels in the top left corner of the low resolution color Cat image, with x and y coordinates both less than 7. As a result, we would expect that in a preimage, there would be some isolated set of color pixels which are never achieved.

To verify this, we need only look at the preimages. If we look at the center of the bird in Figure 6.3c, we can see that the center of the abdomen of the bird is less saturated and has a lighter shade than surrounding pixels. The difference in the performance of the preimage with or without spatial information can be illustrated in how they each deal with that less saturated central patch. Both images seem to correctly associate the pixels in the patch with each other, as evidenced by the pixels within each patch having similar color values. However, the preimage without spatial information fails in predicting the correct hue of the patch, resulting in an unnatural blue patch at that abdominal center.

Figure 6.8: *Comparison of LE preimages of Bird image using either only intensity or intensity* and *spatial information. In (a), we have the preimage of the embedding without spatial information, and the lighter-colored abdomen is assigned the wrong hue. The preimage in (b) is from the embedding which does feature spatial information, and that appears to correct the abdominal pixels to a hue closer to that of nearby pixels. However, this preimage has points outside the pixel value range, with such pixels depicted as black. These pixels are corrected via an averaging post-processing step, shown in (c).*

(a) *LE Pre-image of Bird global rotation w/o spatial*

(b) *LE Pre-image of Bird global rotation with spatial*

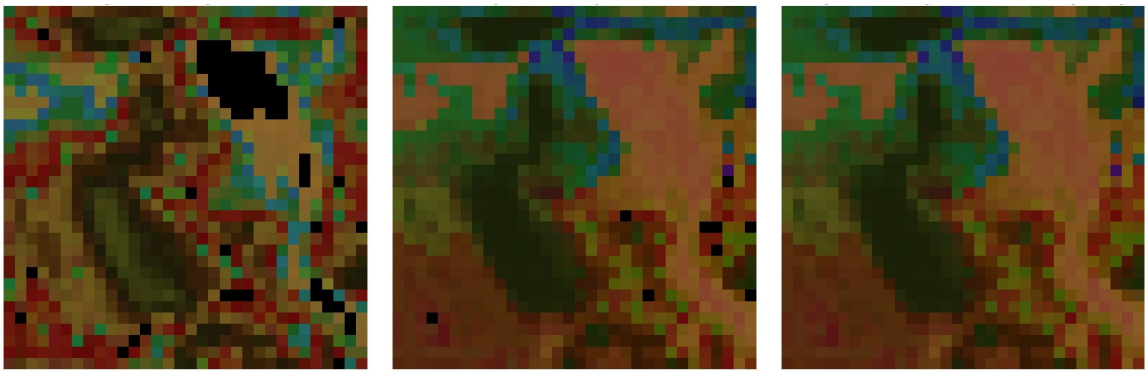
(c) *LE Pre-image of Bird global rotation with spatial and post-processing*

Once the spatial information is included, it becomes easier to reinforce that the pixels in the center of the bird's abdomen should still be associated with the surrounding parts of the bird. Thus, we see a very slight lightening in the center of the abdomen in that preimage. However, the preimage lacks the brightness at that part of the image that is visible in the original image. We believe that this is related to the fact that the boundary of the color points are not reached by the high resolution BW points. This in turn could cause the preimage not to be able to match the full dynamic range of hue and saturation present in the high resolution reference color image. Adding spatial information to the bird also creates some new issues in this instance. Recall that in the final step of the preimage, the pixel coordinates are calculated by finding coordinates which minimize the approximate square distances to its neighborhood set. In particular, there are no additional constraints which force the coordinated into the unit cube $[0, 1]^3$, which is the domain of pixel coordinates. This means that we can obtain preimage pixels with negative values, and such pixels are shown as black on the preimage, or values greater than 1 in a color channel, which results in over saturated pixels that do not look natural.

To remedy the problem caused by preimage pixel vectors which fall outside the unit cube, we implement a simple post-processing correction step. Each preimage pixel with values outside the unit cube is replaced with an equal-weighted average of the preimage values of the up to 8 nearest spatial neighbors that are inside the unit cube $[0, 1]^3$. This provides both a visual improvement in the quality of the preimages and a numerical improvement in PSNR and SSIM.

In the case of Figure 6.9, we see a definite improvement by including spatial information, although the results are still poor in both cases. Note that in the bottom right corner of the original color cat image in Figure 6.4c, there is a patch of green grass. Even though the spatial preimage exhibits less variance among surrounding pixels, both preimages fail to correctly isolate the hue

Figure 6.9: *Comparison of LE preimages of Cat image using either only intensity or intensity* and *spatial information. In (a), we have the preimage of the embedding without spatial information. This preimage has a large amount of heterogeneity in the hue and does not seem to accurately represent the correct image structure. The preimage in (b) is from the embedding which incorporated spatial information, and the hue variation is a closer match to that of the original image. Preimage pixels outside the range of possible pixel values are corrected via an averaging post-processing step, shown in (c).*



(a) *LE Pre-image of Cat global rotation w/o spatial*

(b) *LE Pre-image of Cat global rotation with spatial*

(c) *LE Pre-image of Cat global rotation with spatial and post-processing*

of the bottom right grass patch. Despite this, we believe that these results justify the use of spatial information over excluding it. The remaining preimages in this chapter will therefore use spatial information unless stated otherwise.

6.4.2 Clustered Rotation

Next, we report the result of performing a clustered rotation instead of a global rotation as presented above. That is, we use the method of Section 4.2.2 where the number of clusters P is chosen to be greater that 1. In general, deciding upon the number of clusters in a data set, without any prior knowledge, is its own unsupervised learning task, deserving of separate study. One approach could be use a *hierarchical clustering* method at some fixed linkage to approximate how many clusters are present in the data. Here, we choose based on the variety of visible image

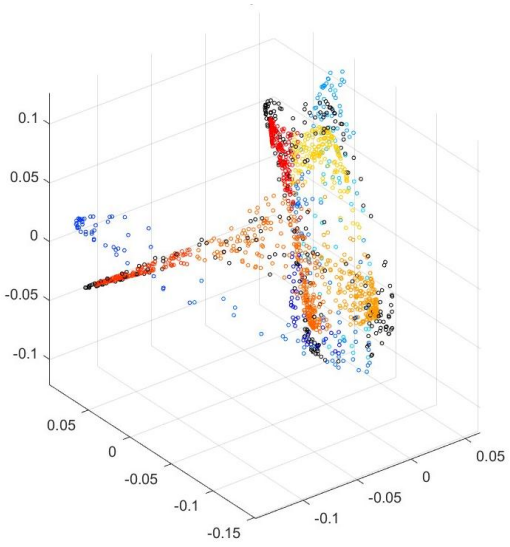
features in the Bird and Cat images, as well as the shape of the embeddings. Choosing 5 clusters for the Bird image and 4 clusters for the Cat image is then a reasonable estimate and sufficient for analysis of the merits of a clustered rotation. For both images, we use both k-means clustering and the curvature-based clustering method developed in Section 4.1 as the clustering methods and perform Procrustes rotations on each cluster.

6.4.2.1 K-means Clustering

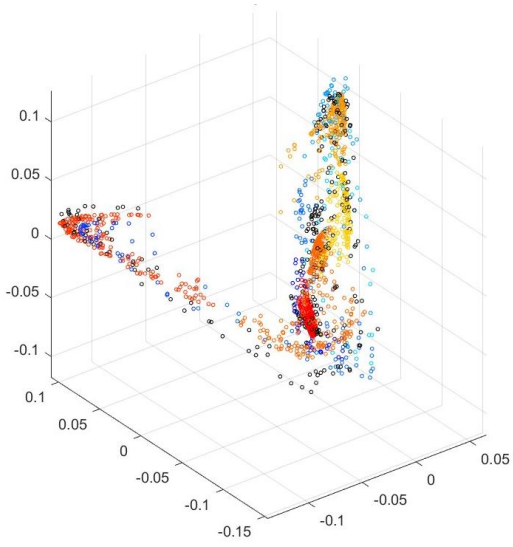
We first study the clusters created using the k-means algorithm on the Bird image. On Figure 6.10a, the cluster in red includes both the upper left portion of the plane-like surface and the peak, away from the flat part of the embedding, at around -0.05 on the z-axis. This is counter-intuitive, as one would expect the points in the left peak to be fairly different to those on the flat portion. Nevertheless, this clustering does not appear to induce a clustered rotation that is much different from the global rotation on the Bird image. There are a few shifts in the shape fitting caused by each cluster rotating separately, but they are minor. The preimage created from the k-means clustered rotation also shares much similarity to that of the global rotation.

The Cat image k-means clustered rotation results in Figure 6.11 also bear much resemblance to the Cat preimage created with the global rotation. There is a slight improvement in matching the blue boundary curve on the right side of the plot in Figure 6.11b when compared to Figure 6.7d. The Cat preimage also appears to be more homogenous in texture at the head of the cat when compared to the other Cat preimages which incorporate spatial information.

Figure 6.10: *Results of 2×2 superresolution on Bird image using k-means clustered rotation. In (a) and (b), low and high resolution BW pixels are represented by black points and points on a color gradient of yellow to red, respectively. Low resolution color pixels are represented by points on a blue gradient. Different colors within a color gradient represent different cluster assignments. (a) shows both LE embeddings on the same plot, and (b) depicts the result of the k-means clustered rotation on the BW embedding into the color embedding space. Compared to Figure 6.7b, the rotation in (b) matches the color points more closely. (c) shows the resultant high resolution color preimage after k-means clustered rotation. This preimage still suffers from the same out-of-range issue as Figure 6.8b.*



(a) *Bird image Before rotation*

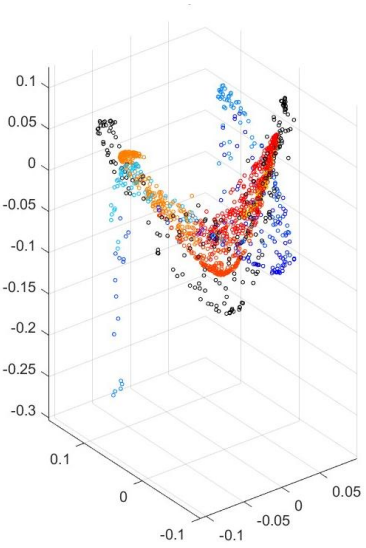


(b) *Bird image After rotation*

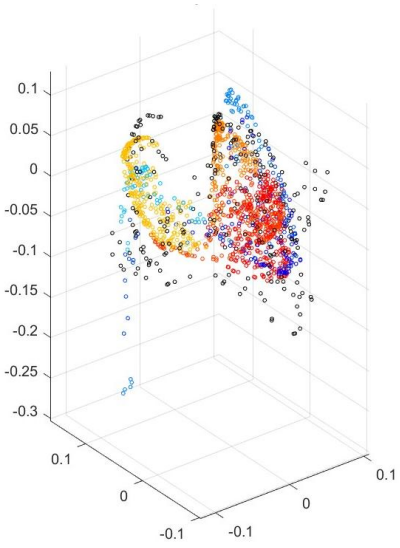
(c) *LE preimage of Bird image after k-means clustered rotation*

(d) *LE preimage of Bird image after k-means clustered rotation and post-processing*

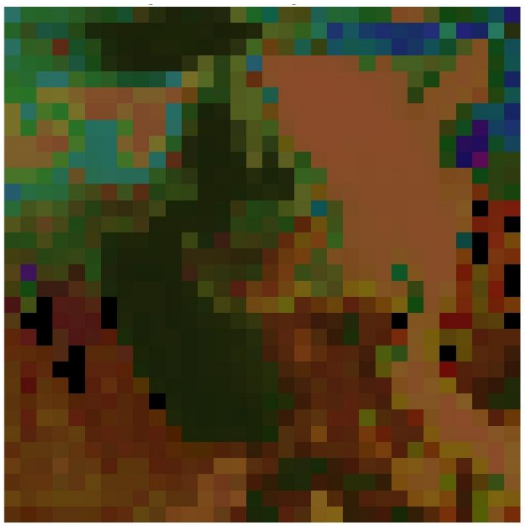
Figure 6.11: *Results of 2×2 color MMSR on Cat image using k-means clustered rotation. In (a) and (b), low and high resolution BW pixels are represented by black points and points on a color gradient of yellow to red, respectively. Low resolution color pixels are represented by points on a blue gradient. Different colors within a color gradient represent different cluster assignments. (a) shows both LE embeddings on the same plot, and (b) depicts the result of the k-means clustered rotation on the BW embedding into the color embedding space. There is still not a cluster of BW pixels in (b) which matches the set of color points going down the z-axis, previously noted in Figure 6.7c. (c) shows the resultant high resolution color preimage after k-means clustered rotation. This preimage does not appear to be an improvement over the global rotation preimage in Figure 6.9b.*



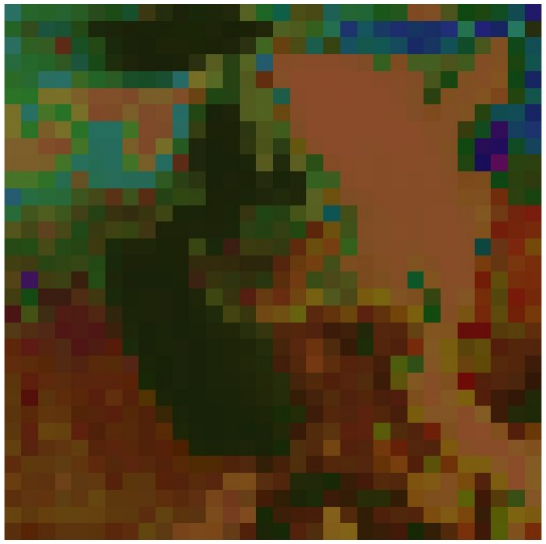
(a) *Cat image Before rotation*



(b) *Cat image After rotation*



(c) *LE preimage of Cat image after k-means clustered rotation*



(d) *LE preimage of Cat image after k-means clustered rotation and post-processing*

6.4.2.2 Curvature-based Clustering

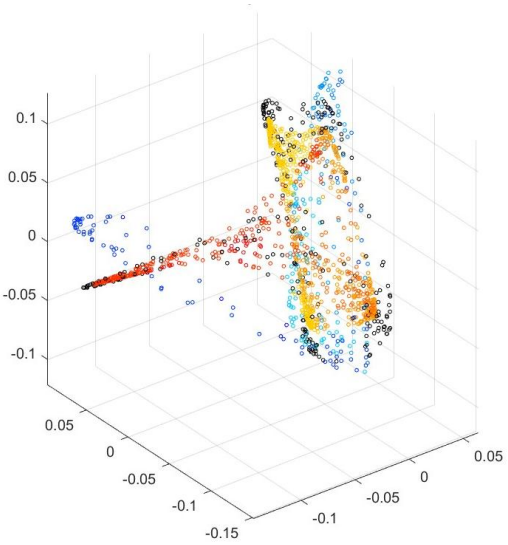
As demonstrated in Section 4.1, k-means curvature can sometimes fail to create clusters which are close to being affine. As our clustered rotation method uses rotations to map between LE embeddings, the shape of the cluster is paramount to successfully matching the shapes of the embeddings, and ultimately, to the quality of the preimage. Next, we test whether the curvature-based clustering method provides better rotation and preimage results for the Bird and Cat images.

On the Bird image, curvature clustering is able to separate clusters into reasonable geometric regions, as the left and right sides of the flat plane-like part in Figure 6.12a are both separated from the peak to the left of the plot, and all in different clusters. Despite this, the clustered rotation does not differ much from that of the k-means clustering, in terms of how the BW pixels, points in the red-to yellow gradient in Figure 6.12, fill up the LE embedding space of color pixels, denoted in blue. The preimage also has pixels outside the correct range, as evidenced by incorrect black pixels and certain highly saturated pixels, like a turquoise one at the top of the bird's front leg in Figure 6.12c.

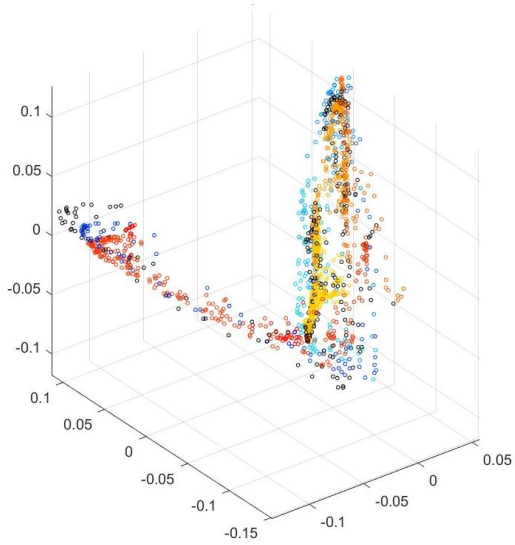
A drawback in the curvature clustering method is that it does not have any constraints to enforce cluster size uniformity. This can result in clusters which consist of only a few points, as is the case for the Cat image in Figure 6.13. The majority of the BW embedding is contained in one cluster, depicted in orange in the figure. This means that we would expect the results here, both in terms of the derived mapping between embedding spaces and the quality of the resultant preimage, to not be very different from those in the global rotation case.

We also note that all the rotation methods, when applied to the Cat image, have failed to

Figure 6.12: *Results of 2×2 color MMSR on Bird image using curvature-based clustered rotation. In (a) and (b), low and high resolution BW pixels are represented by black points and points on a color gradient of yellow to red, respectively. Low resolution color pixels are represented by points on a blue gradient. Different colors within a color gradient represent different cluster assignments. (a) shows both LE embeddings on the same plot, and (b) depicts the result of the curvature-based clustered rotation on the BW embedding into the color embedding space.*



(a) *Bird image Before rotation*

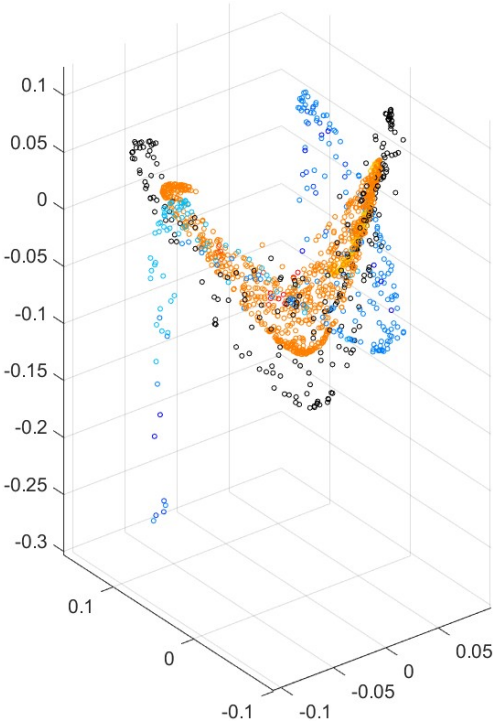


(b) *Bird image After rotation*

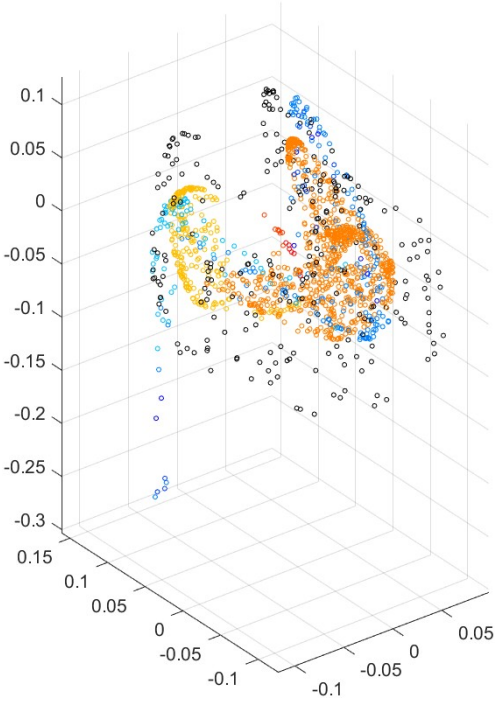
(c) *LE preimage of Bird image after curvature-based clustered rotation*

(d) *LE preimage of Bird image after curvature-based clustered rotation and post-processing*

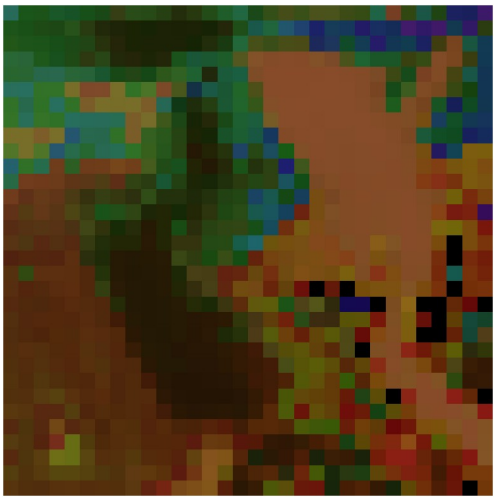
Figure 6.13: *Results of 2×2 color MMSR on Cat image using curvature-based clustered rotation. In (a) and (b), low and high resolution BW pixels are represented by black points and points on a color gradient of yellow to red, respectively. Low resolution color pixels are represented by points on a blue gradient. Different colors within a color gradient represent different cluster assignments. (a) shows both LE embeddings on the same plot, and (b) depicts the result of the curvature-based clustered rotation on the BW embedding into the color embedding space.*



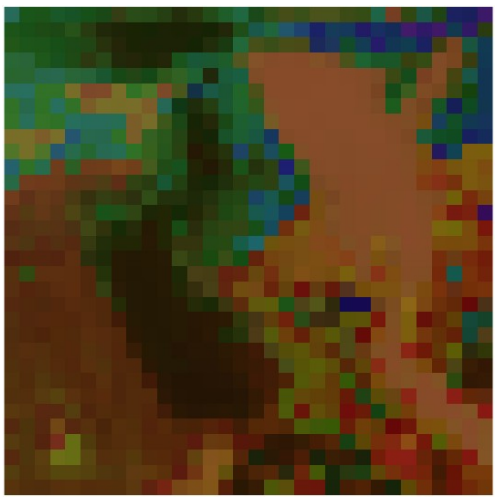
(a) *Cat image before rotation*



(b) *Cat image after rotation*



(c) *LE preimage of Cat image after curvature-based clustered rotation*



(d) *LE preimage of Cat image after curvature-based clustered rotation*

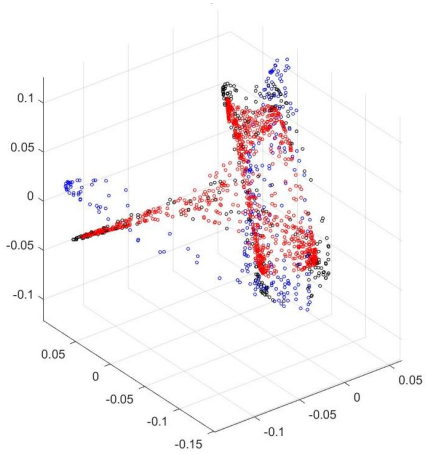
reconstruct the patch of grass located in the bottom right corner of the reference high resolution image in Figure 6.4c. We attribute this to certain parameter choices in the graph construction. There are only about 20 or so pixels of grass in the low resolution color image, so for a high resolution BW grass pixel, a set of 60 nearest neighbors will never contain a majority of grass pixels. On the other hand, having sparser neighborhoods with fewer neighbors decreases the overall connectivity of the graph, which expands the scale of the embedding. This, in turn, can cause issues with the mapping between embedding spaces. A potential solution could be to increase the number of eigenvectors used for the LE embedding, as it has been shown that the size of clusters in an LE embedding is inversely related to the number of eigenvectors needed for representation on that cluster [27]. Investigating this further is then reserved for future research projects.

6.5 Color MMSR By Graph Matching

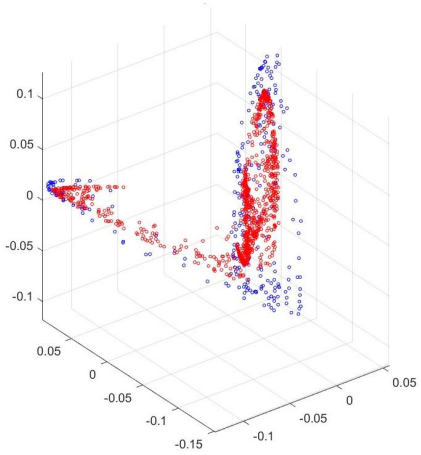
Our final experiment in color MMSR under the data fusion model involves using the graph matching approach to form the mapping between BW and color LE embedding spaces. In this situation, high resolution BW pixels are directly embedded into the color embedding space, based on the relationships the high resolution BW pixels and the low resolution BW pixels which are identified with the low resolution color pixels. We would then expect that this out-of-sample extension is able to spread the high resolution BW pixels in the color LE embedding as they are in the BW LE embedding. We will analyze the results of this method on the Bird and Cat images, and conclude the chapter with a table of PSNR and SSI values, Table 6.1, for the different preimage approaches considered so far. In Figures 6.14 and 6.15, low resolution color pixels are given as blue points, and high resolution BW pixels are given as red points. The low resolution BW pixels are not explicitly shown after the matching as they are sent to the exact locations of their corresponding color pixels.

To analyze the quality of the graph matching approach on the Bird image, we first compare the matched high resolution BW pixels in Figure 6.14b to some of the other rotations from the previous section. One of the clearer differences, compared to the clustered rotations in particular, is a stricter adherence to staying within the convex hull of the blue embedded points. This makes sense, as graph matching creates a mapping between the spaces that is not encumbered by the rigidity of the Procrustes rotation. On the other hand, this creates a perimeter of color points which are not reached by the BW points, which could negatively affect the quality of the preimage. Indeed, the preimage still suffers from pixels vectors landing outside the unit cube. As this occurs in both preimage methods, this is likely an issue in the preimage itself, specifically

Figure 6.14: *Results of 2×2 color MMSR on Bird image using graph matching. In (a) and (b), high resolution BW pixels are represented by red points and low resolution color pixels are shown as blue points. Here, the low resolution BW and color pixels are identified, so the low resolution BW points are not shown. The color preimage of the Bird from the graph matching approach is shown in (c).*



(a) *Bird image before matching*



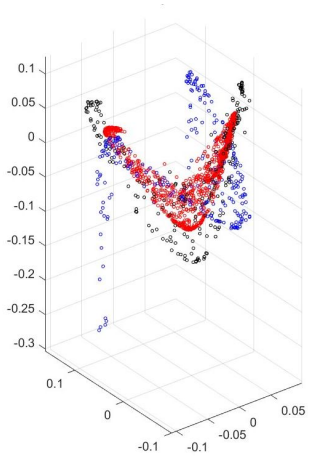
(b) *Bird image after matching*

(c) *LE preimage of Bird image after matching*

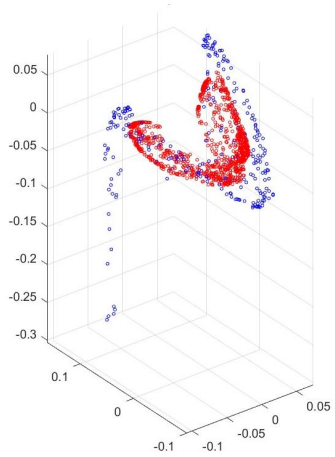
(d) *LE preimage of Bird image after matching and post-processing*

the optimization step which finds an approximate kernel vector. An incorrect choice affects both the approximated neighbors and their distances, which can result in poor performance.

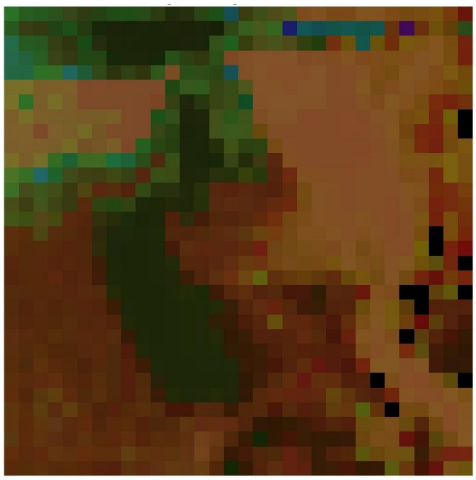
Figure 6.15: *Results of 2×2 color MMSR on the Cat image using graph matching. In (a) and (b), high resolution BW pixels are represented by red points and low resolution color pixels are shown as blue points. Here, the low resolution BW and color pixels are identified, so the low resolution BW points are not shown. The color preimage of the Cat from the graph matching approach is shown in (c).*



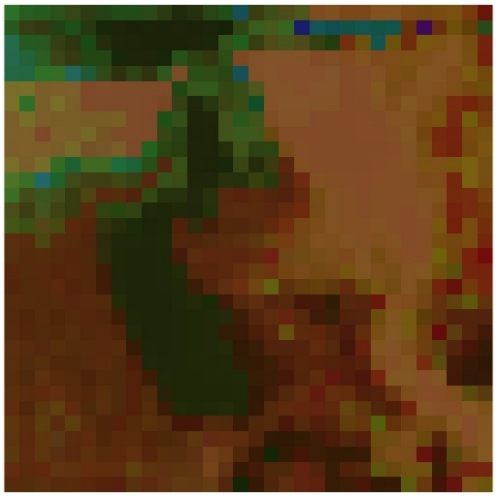
(a) *Cat image before matching*



(b) *Cat image after matching*



(c) *LE preimage of Cat image after matching*



(d) *LE preimage of Cat image after matching and post-processing*

We can see the results of graph matching applied to the Cat image in Figure 6.15. The matched BW pixels seem to "fold" along the main curve of the color pixels in Figure 6.15b compared to the rotations, both clustered and global, which are more slightly congested near the

center of the color embedding. Isolation of the trail of color points in the figure persists, which supports the conjecture that this is due to graph construction more so than a failure in the mapping between the points. This further elucidates the importance of parameter selection in our color MMSR algorithm, and in the data fusion model in general. The preimage after graph matching shows improvement over those of the rotations. The colors on the cat appear more continuous with respect to the spatial location of each pixel, as opposed to the 2 clustered rotations which demonstrate heterogeneity of hue on the head of the cat.

6.6 Numerical Results and Conclusion

6.6.1 PSNR and SSIM Results

We present then results of the 5 different preimage experiments conducted, both with and without the post-processing step applied to preimage color vectors which fall out of the unit cube.

First, the graph matching approach is very clearly the superior method for both images and across both metrics. This is a good result because the graph matching method does not require the extra parameter of number of clusters. In the first testing of the curvature-based clustering method, we see that it can outperform k-means clustering in certain situations, such as the LE embedding of the low resolution BW Bird image. However, the two clustering methods perform worse than or about equal to the global rotation. Further study is required to see how these results change with the number of clusters, or with other clustering methods like hierarchical clustering. For the Cat image, curvature clustering actually does worse than not having spatial information at all in terms of SSIM. We attribute this to the fact that the method is currently dependent on the choice of 2-dimensional projection. When developing the method, we noticed similar incidents in which one or a small few clusters contained a disproportionate number of points. This would sometimes change if a different pair of eigenvectors was chosen to project on. We would ultimately like to adapt this method in the future to lose this dependence on a particular choice of projection.

Note that for the spectral-only global rotation Bird preimage, the post-processing step does not change the results. This is because that spectral-only global preimage does not actually have any pixels outside the unit cube; the light blue pixels in the abdomen of the bird in Figure 6.8a

Table 6.1: *PSNR and SSIM values for Bird and Cat preimages. (a) shows the values for the images before the post-processing step, and (b) shows the values after cleaning up the preimage. PSNR values take up the top half of each line, while SSIM are on the bottom. The largest PSNR or SSIM value across all methods is denoted in* ***bold****. Graph matching produces the best results for both images, but the performance gap is slightly wider for the Cat image. The curvature-based clustering performs poorly on the Cat image, but better than k-means on the Bird image.*

		k-means rotation	Spatial + spectral global rotation	curvature rotation
Bird	PSNR	22.21	22.33	22.35
	SSIM	0.898	0.903	0.901
Cat	PSNR	18.74	18.69	18.42
	SSIM	0.901	0.900	0.89

		Spectral global rotation	Graph matched
Bird	PSNR	20.89	**22.73**
	SSIM	0.886	**0.912**
Cat	PSNR	17.87	**21.71**
	SSIM	0.887	**0.952**

(a) *Results without post-processing*

		k-means rotation	Spatial + spectral global rotation	curvature rotation
Bird	PSNR	22.30	22.58	22.47
	SSIM	0.899	0.908	0.903
Cat	PSNR	18.77	18.70	18.42
	SSIM	0.901	0.900	0.891

		Spectral global rotation	Graph matched
Bird	PSNR	20.89	**22.81**
	SSIM	0.886	**0.913**
Cat	PSNR	18.26	**21.75**
	SSIM	0.893	**0.952**

(b) *Results with post-processing*

are inaccuracies, but still fall within the feasible region of pixel vectors.

6.6.2 Discussion

We set out to create a color transformation and superresolution method in order to demonstrate the potency and generality of the data fusion model. Through this process, we saw how parameter selection for the graph construction can be crucial to results. Applying this method on small data sets such as these images, consisting of at most $32^2 = 1024$ points at a time, allowed for detailed analysis of the factors that contribute to a visually appealing and contextually relevant preimage. For instance, when we originally designed the experiment, our first idea was to directly embed the high resolution BW pixels without the use of an out-of-sample extension. From there, we could create rotations anchored by the choice of a sampling of high resolution points that would correspond to the low resolution points in the color LE embedding. However, we realized that the disparity in the number of points in each embedding created a scale discrepancy that was insurmountable. Using the out-of-sample extension avoids this issue and aids in scale matching, as the same number of points generate each LE embedding. In addition to scale, we also learned that large shape differences can inhibit construction of suitable mappings between embeddings. A clear example of this kind of shape mismatch can be found in all the embeddings using the Cat image, as there was always a distinct set of color points which were not approached by BW points after a mapping.

Ultimately, we conclude that there is validity in the use of the data fusion model towards this problem, both in terms of color transformation and superresolution. However, our current algorithm does not compare favorably with other methods which exploit physical models to bet-

ter recover color information [4, 51, 92]. Instead, our work is notable as an extension of the established literature involving the use of graph Laplacians for color transformation. In [51], Eynard, Kovnatsky, and Bronstein address the color transfer problem, not including superresolution, through use of a joint embedding space, created with a *joint approximate diagonalization* of two graph Laplacians representing different color gamuts.

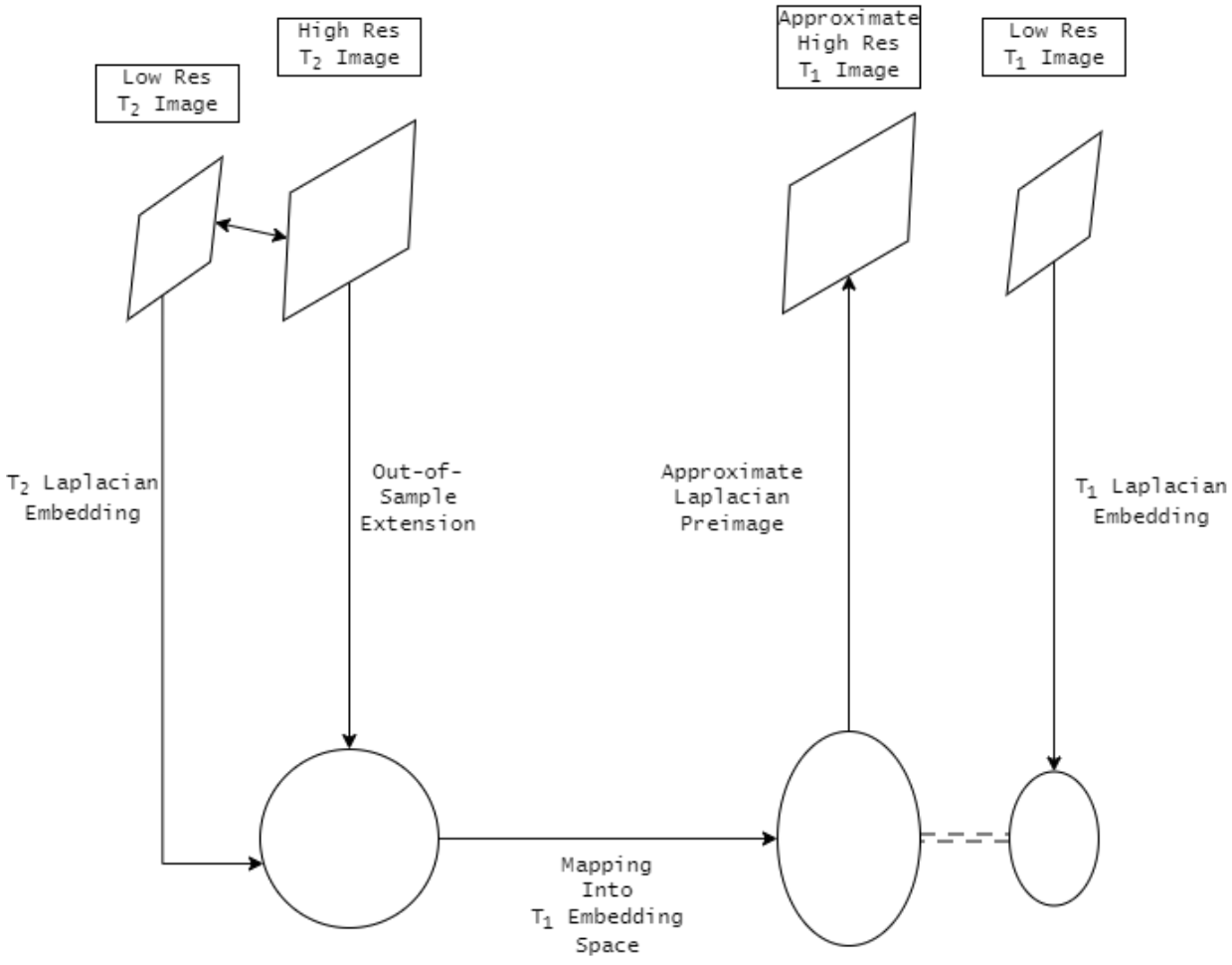
In the following chapter, we apply the lessons learned through this study towards a multimodal superresolution technique for MRI, based on the graph Laplacian data fusion model.

Chapter 7: Multimodal Superresolution in MRI

The basic idea is to create a high resolution T1w image using a low resolution T1w image and a high resolution T2w image. This is what we refer to as the multimodal superresolution (MMSR) problem for MR images. This is motivated by differences in acquisition time of the two MR modalities, as there are instances in which the acquisition time for a T2w image is shorter than that of a T1w image. In this chapter, we develop a method for the MMSR problem for MR images.

Given a high resolution T2w image and a low resolution T1w image of the same region, we seek a high resolution T1w image. Our approach to this problem is to use the graph Laplacian data fusion model described in Section 4.3. First, we create a low resolution T2w image, with the same resolution as the low resolution T1w image, from our high resolution T2w image. This means we can identify pixels at the same spatial location on both low resolution images with each other. Then, we compute Laplacian eigenmap (LE) embeddings of both low resolution images, forming T1w and T2w embedding spaces. We then use the Nyström extension to embed the high resolution T2w pixels into the T2w space. From there, we map the T2w points into the T1w embedding space using one of the methods of Section 4.2. Finally, the LE preimage of Section 3.6 is applied to the mapped T2w points, which gives a high resolution image with similar image features as a T1w image. Our method is illustrated in Figure 7.1.

Figure 7.1: A diagram of the MRI MMSR algorithm



Each data point represents the tissue contained in one pixel, or, more accurately, each voxel, represented by the T1- or T2-weighted proton density as image intensity. Despite the maturity of MR as a technique, the relationship between modalities is much harder to define than that of grayscale and color images. In general, there is no fixed relationship between the T1 and T2 constants across all objects. For tissues found in the brain though, there is roughly a one-to-one relationship between T1 and T2, made evident by considering the molecular mobility of the molecules in such tissues. [19]. In light of this, we believe that the problem of recovering T1w pixel information from T2w images, specifically in the context of MR brain scans, is feasible.

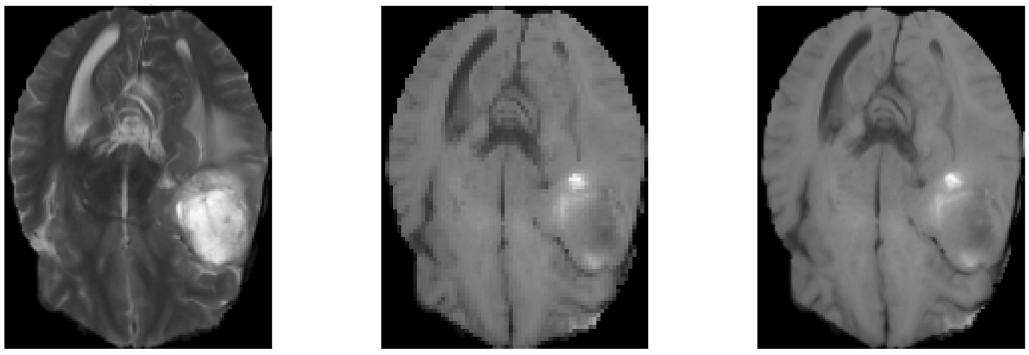
We use two data sets here, both of which are publically available and consist of MRI brain scans: the Brain Tumor Segmentation (BraTS) Challenge 2017 data set [6, 7, 88] and the BrainWeb Simulated Brain Database [33, 71, 72]. The BraTS data set contains actual, clinical MRI scans of tumor patients whereas the BrainWeb data set is made of simulated images from an anatomical model. Therefore, we place more value on the development and results of our method on the BraTS images. However, BrainWeb images provide a useful benchmark for comparing images and allows for testing of the method without the difficult pathologies of BraTS images, which can sometimes complicate image analysis.

We specifically use two imaging volumes from the BrainWeb data set as these simulated images are not very varied, the BrainWeb normal volume and the BrainWeb MS volume, which models a brain with lesions due to multiple sclerosis (MS). The BraTS volumes are preprocessed to remove unneeded image features, such as fat around the skull, and are resized such that each voxel covers a volume $1mm$ each in width and length and $5mm$ in depth. The BrainWeb volumes have voxels which are $1mm^3$, $1mm$ long in each direction, and contains fat and other tissue around the skull. As a preprocessing step, we remove this surrounding tissue using the 3D Slicer

image computing software [52]. As the MR data is presented as a volume, we have the option to choose images along 3 planes, but we choose slices *in-plane*, in *axial* view unless otherwise stated. In the axial view, slices are perpendicular to an imagined line from the foot to the head.

Our low resolution images are all created by taking linear averages of the high resolution images available in the dataset in use. For positive integer $r \geq 1$, we say that a low resolution image is an $r \times r$ *downsizing* of a high resolution image if both of the spatial dimensions of the high resolution image are scaled up from their corresponding low resolution dimensions by r, and we refer to r as the *scaling factor*. In the experiments described in this chapter, we consider scaling factors $r = 2, 3$ and 8.

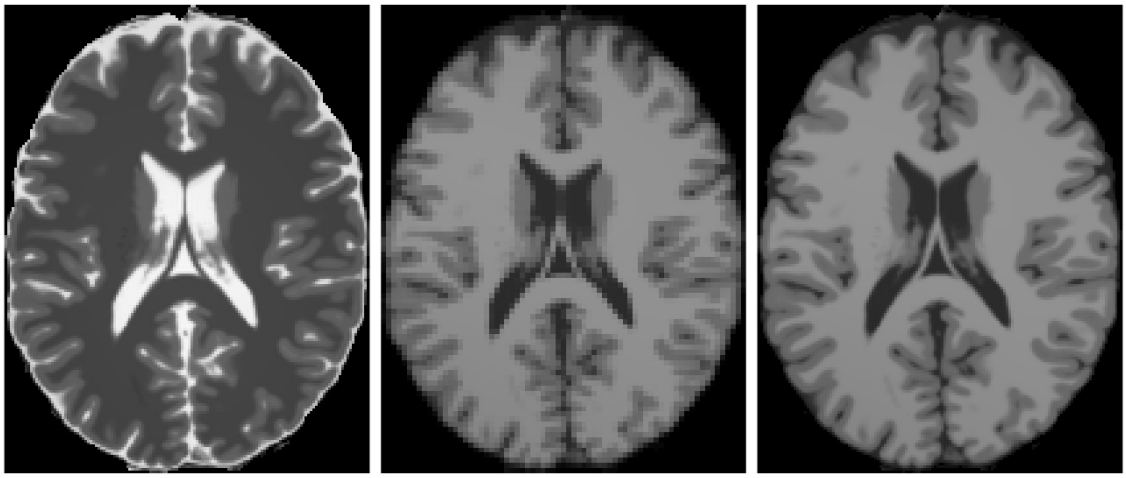
Figure 7.2: *Example axial slice of an BraTS volume* 1*, slice* 68*. The high resolution images (a) and (c) are* 172 *pixels in the vertical direction and* 134 *pixels in the horizontal direction, while the low resolution image (b) is scaled down by scaling factor* $r = 2$ *in both dimensions, resulting in an image resolution of* 86 *pixels by* 67 *pixels.*



(a) *High Resolution T2w Image* **(b)** *Low Resolution T1w Image* **(c)** *High Resolution T1w Image*

Figure 7.2 shows an example image of from a brain MR volume from the BraTS 2017 challenge data set [87]. This example slice of the full MR volume is cropped to 172×134 pixels. We omit the background black pixels from the graph construction, leaving 18129 pixels as T2w high resolution points. The same reduction leaves 4529 pixels of the 67×86 pixel low resolution T1w image, downsized by a scaling factor of 2, to form a T1w LE embedding. Likewise, 7.3

Figure 7.3: *Example axial slice, BrainWeb normal volume, slice* 91*. The high resolution images (c) and (a) are* 181 *pixels in the vertical direction and* 141 *pixels in the horizontal direction, while the low resolution image (b) is scaled down by scaling factor* $r = 2$ *in both dimensions, rounded to* 91 *pixels by* 71 *pixels. BrainWeb images are synthetic and thus exhibit less detail than BraTS images.*



(a) *High Resolution T2w Image* **(b)** *Low Resolution T1w Image* **(c)** *High Resolution T1w Image*

shows an example from the BrainWeb normal volume. The image is cropped to 181×141 pixels in high resolution and 91×71 pixels in low resolution. Once the black background pixels are omitted, there are 19829 T2w high resolution points and 5127 low resolution pixels of both modalities used to construct the two embedding spaces. Figure 7.2 in particular demonstrates how the two different MR modalities highlight the same physical features in contrasting ways.

7.1 Parameter Selection

The experiments of Chapter 6 demonstrate how important proper parameter selection is for Laplacian eigenmap embedding spaces to be sufficiently representative of image features. In this section, we describe some considerations made in our choice of LE embedding parameters.

7.1.1 Kernel bandwidth parameter σ

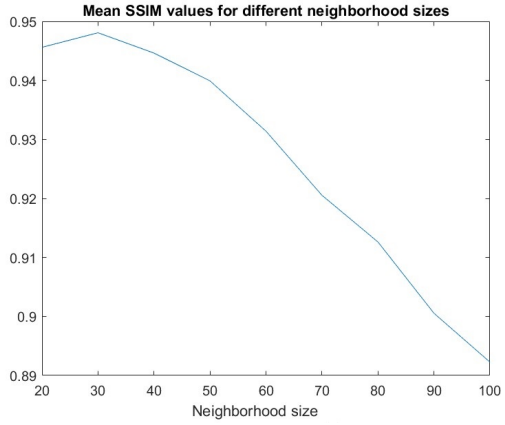
Because we choose the standard Gaussian weight function $w(x, y) = exp(-\frac{\|x-y\|^2}{2\sigma^2})$ along with a nearest neighbor condition, σ is a *kernel bandwidth* parameter which controls the strength of points which are already connected. In the ideal case, this bandwidth parameter allows the weight function to prioritize strong local connections. The nearest neighbor condition already does the task to an extent, but as we require the graph adjacency matrix to be symmetric, neighborhood sets for a point can contain some weaker connections. Because of this, we want to choose σ in a way that prioritizes the stronger connections, that is, data points which are closer in Euclidean distance. Lindenbaum et al. [81] detail ways in which σ is chosen in the literature, some of which allow for varying σ values throughout the data set, but we choose to use a simple statistic of the measured distances between connected points. For our experiments, σ is chosen to be the 80-th percentile of all distances between neighboring pixel vectors. This is an adaptation of the common decision to base σ on the empirical distribution of squared distance in the data [81].

When we take the pixel vectors of an image, without considering the spatial locations of pixel on the image, then this is a suitable choice. However, we do want to consider the use of spatial information in creating pixel vectors, as detailed in Section 4.4.1.1. This requires a rescaling of spatial coordinates in order to control the influence of spatial information. Our experiments are conducted using a scaling factor μ of $\frac{1}{5}$ unless otherwise stated. This means that while the intensity values of any pixel can range in the interval $[0, 1]$, the spatial coordinates are restricted to $[0, \mu]^2$.

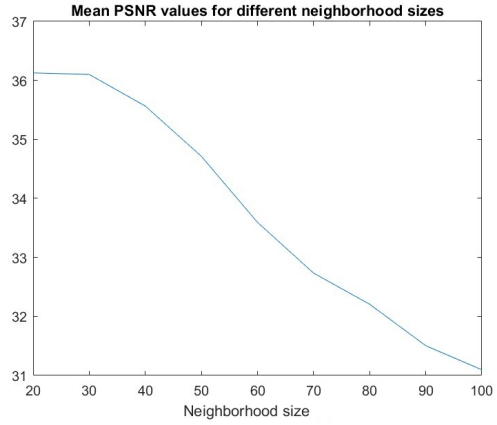
7.1.2 Neighborhood Size and Embedding Dimension

The neighborhood size for each point is usually the largest contributor to the connectedness of the graph, which in turn controls how well the first few eigenvectors are able to capture data features. Related is the number of eigenvectors used; increasing this generally allows for the representation of more subtle data features which are localized in embedding space [27].

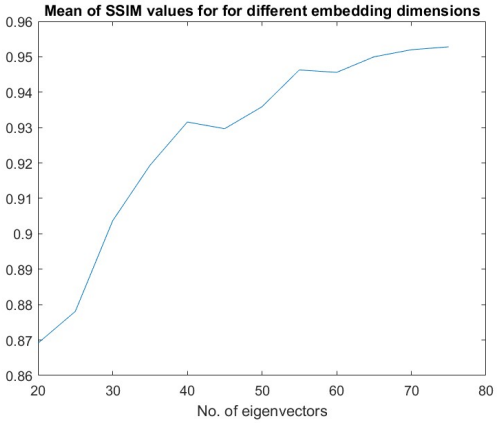
Figure 7.4: *4 graphs of mean SSIM and PSNR metrics for different choices of LE embedding parameters. Each metric measures the accuracy of recovering an image from its LE embedding via the approximate preimage. This experiment is conducted over 5 image slices, using 9 choices of neighborhood size (m) and 12 choices of embedding dimension d. The general trend seems to be that increasing dimension and keeping the neighborhood size low leads to improved image recovery.*



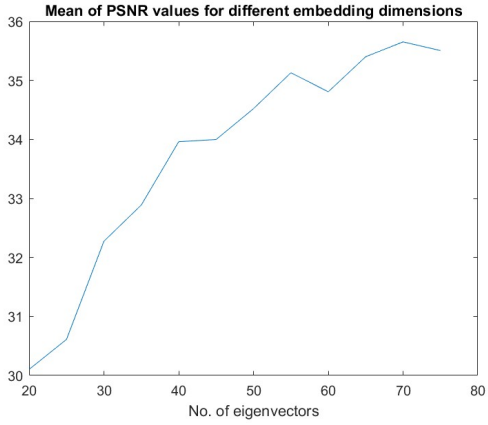
(a) *Mean SSIM for different neighborhood sizes*



(b) *Mean PSNR for different neighborhood sizes*



(c) *Mean SSIM for different LE dimensions*



(d) *Mean PSNR for different LE dimensions*

As a means of establishing a set of parameters which are suitable in our embedding constructions, we implement a simple grid search, testing all combinations over a range of choices of neighborhood size and the number of eigenvectors used in creating the Laplacian eigenmaps embedding. We evaluated these options by performing a near "identity" transformation for a cropped version of a T1w image. With varying neighborhood sizes, the set of measured distances between connected points at each trial changes. This means that we need to select a fixed σ for this test, instead of using the method described in the previous section. We take $\sigma = 0.01$ for this test, as we found that smaller orders of magnitude result in unconnected graphs.

Our experiment consists of evaluating the embeddings of images cropped to 90×84, meaning 7560 pixels are used to create each embedding space. We select slices near to the center of the brain, containing most of the important image features such as gray matter, white matter, and tumor, and our crop is chosen in the center of each slice. Each is embedded into its LE embedding space, and then the preimage is directly calculated. Ideally, this process should exactly reproduce the test image. We then measure how close the preimage is to the original using peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM), defined at the start of Chapter 6. We include PSNR as it is commonly found in literature related to superresolution and image processing problems, but we place more importance on SSIM in choosing parameters due to its ability to measure accuracy of image structure and contrast and because of its relation to the quality perception of the human visual system [65].

Our test images are taken from the BraTS data set. We select one slice from volume 5, two slices from volume 27, and the same slice twice, with different spatial rescaling factors μ, from volume 1. For each slice, we create 108 LE embeddings, using 12 values of embedding dimension d, $d = 25, 30, 35, \ldots, 75$ and 9 values of neighborhood size m, $m = 20, 30, 40, \ldots, 100$.

Then, fixing either m or d, we compute the mean score for each of the metrics over all 60 or 45 experiments, respectively, with that choice of m or d. The graphs from this analysis are shown in Figure 7.4.

Figure 7.4 shows that across different slices and choices of embedding dimension, best results in both metrics occur for $20 \leq m \leq 50$. It is also apparent that $d \geq 50$ has better performance. However, d affects the computational cost of the algorithm considerably more than m does, as m is only incorporated in the initial construction of the graph Laplacian, while the transformations which occur in, and coming out of, the embedding space are all determined by the dimension d. For this reason, we are incentivized to keep d small without too much of a drop in either metric.

At this point, further analysis of the results are required for each of the slices individually. In Table 7.1, we present the full set of SSIM and PSNR values for all 108 experiments for BraTS volume 1, slice 68 using spatial rescaling factor $\mu = \frac{1}{5}$. Guided by Figure 7.4, we select the pair (m, d) within the range $20 < m < 50$, $d > 50$ with smallest value of d which serves as a local maximizer of SSIM. PSNR is not used for this selection because we prioritize visual quality of the preimage over noise reduction. Under these criteria, we find that $(m, d) = (40, 60)$ gives a local maximum value of 0.964. Notice though that the highest value of SSIM occurs at $d = 75$, and there is another local maximizer at $d = 70$, but the general trend indicates that we are likely to continue getting better results for larger values of d. This procedure for parameter selection provides a stopping point for d that results in stable, high quality embeddings.

At such a local maximizer, we still obtain results which are close to the optimal mean SSIM. Figure 7.4a shows that the mean SSIM is maximized at $n = 30$ at 0.948, but the value at $m = 40$ is within 0.36% of the maximum at 0.945. Similarly, Figure 7.4c shows the mean SSIM

Table 7.1: *Example table of parameter values for the* identity *transformation with $\sigma = 0.01$ on a slice from Volume 1 of the BraTS data set. The top table (a) shows SSIM values for different combinations of number of eigenvectors d and neighborhood size m, while the bottom table (b) does the same for PSNR. Best results in both tables are between $m = 20$ and $m = 50$ and $d > 50$, with the maximum shown in* ***bold****. The* local *maximizer of SSIM with smallest d within that range on (a) is $d = 60, m = 40$, shown* underlined*. Using a local maximizer provides a high quality LE embedding without having to incur the increased computational cost of a large value of d.*

$m =$	20	30	40	50	60	70	80	90	100
$d = 20$	0.881	0.871	0.866	0.839	0.831	0.821	0.803	0.791	0.788
$d = 25$	0.888	0.868	0.867	0.886	0.888	0.874	0.856	0.852	0.796
$d = 30$	0.918	0.927	0.928	0.922	0.918	0.867	0.863	0.866	0.830
$d = 35$	0.939	0.920	0.914	0.922	0.910	0.892	0.898	0.884	0.872
$d = 40$	0.930	0.946	0.948	0.943	0.938	0.868	0.909	0.916	0.908
$d = 45$	0.940	0.954	0.952	0.944	0.930	0.937	0.913	0.884	0.861
$d = 50$	0.949	0.958	0.955	0.955	0.950	0.942	0.931	0.925	0.915
$d = 55$	0.954	0.942	0.958	0.957	0.951	0.944	0.937	0.936	0.920
$d = 60$	0.955	0.963	0.964	0.962	0.949	0.949	0.944	0.930	0.932
$d = 65$	0.958	0.964	0.961	0.961	0.952	0.951	0.944	0.940	0.927
$d = 70$	0.954	0.966	0.967	0.964	0.952	0.956	0.953	0.950	0.930
$d = 75$	0.960	0.963	0.964	**0.968**	0.946	0.937	0.942	0.937	0.916

(a) *SSIM values*

$m =$	20	30	40	50	60	70	80	90	100
$d = 20$	32.45	31.50	31.04	30.09	29.74	29.17	28.14	27.54	27.12
$d = 25$	32.86	31.20	31.24	32.53	32.12	31.30	30.65	30.51	28.68
$d = 30$	34.56	35.42	35.08	34.11	33.64	30.22	29.87	30.56	29.26
$d = 35$	36.49	34.66	32.95	34.38	33.40	31.97	32.50	31.85	30.90
$d = 40$	35.73	37.14	36.96	36.16	35.66	31.21	33.20	33.33	33.19
$d = 45$	36.69	37.67	37.21	35.92	33.03	34.80	33.14	31.79	30.78
$d = 50$	37.66	38.10	37.43	37.31	36.60	35.79	34.56	34.05	33.56
$d = 55$	38.26	35.70	37.90	37.51	36.78	35.80	35.10	35.06	33.94
$d = 60$	38.48	38.85	38.56	37.89	36.09	35.79	35.35	34.42	34.36
$d = 65$	38.73	38.83	38.47	37.73	36.58	36.36	35.87	35.30	34.43
$d = 70$	38.37	**39.16**	38.92	38.03	36.75	36.69	36.23	35.35	33.93
$d = 75$	38.95	38.12	36.79	38.34	33.46	34.02	34.36	33.86	32.44

(b) *PSNR values*

values for $d \geq 55$ are all within 0.76% of the maximum mean of 0.953, achieved at $d = 75$. Thus, we recommend this process for choosing suitable values of neighborhood size m and embedding dimension d for each image. Unless otherwise stated, these parameters, $(m, d) = (40, 60)$, are used for the subsequent computations in this chapter. We emphasize that the selection of optimal parameters for kernel methods such as Laplacian eigenmaps is a difficult problem, even without accounting for varying such parameters across many different images. The purpose of this experiment is not to find the *optimal* parameters for our process, but to select stable parameters for faithful embeddings across multiple images and modalities.

7.2 Spatial-Spectral Information in Images

Once we have a reasonable range for graph construction parameters, it is important to decide what information is used to create the LE embedding. In this section, we explore the results of including or excluding spatial information and justify our decision to use the spatial data.

First, from a theoretical point-of-view, it seems wasteful *not* to incorporate spatial information in this context, as these MR images depict physical locations in the brain. As natural images, MR images exhibit local spatial structure. Further, the LE embedding and preimage methods have difficulty with large subsets of identical data points; this is, in part, why black background pixels are not included in the graph construction and subsequent calculations. The addition of spatial data, done in the way described in Section 4.4.1.1, guarantees that no two pixels have the same point in the LE embedding.

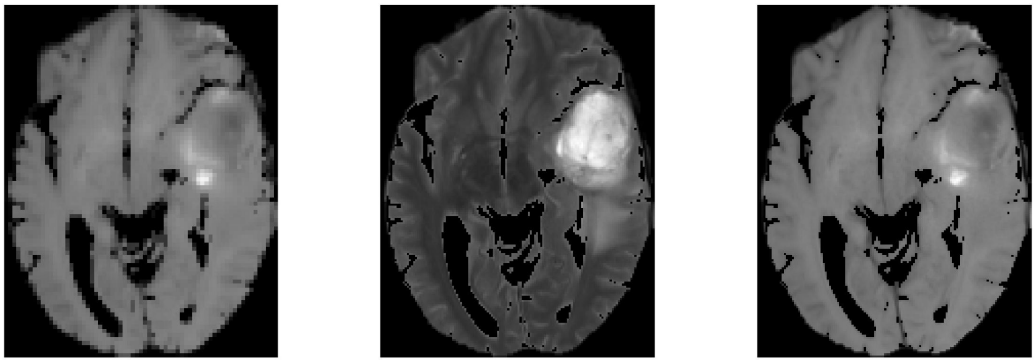
We then conduct a short experiment to evaluate the two options. Using BraTS volume 1,

slice 68, we run the 2×2 MMSR procedure twice, once with spatial information and another without, to obtain a high resolution T1w image from a high resolution T2w image and a low resolution T1w image. The pixel coordinates when used are scaled to form a lattice in $[0, \frac{1}{5}]^2$ and all other parameters are the same: 40-nearest neighbors, 60 Laplacian eigenvectors, and σ determined as the 80-th percentile of all distances between neighboring pixel vectors. Once the LE embeddings are created, we use the graph matching approach of Section 4.2.3 to map from the T2w space to the T1w space, and then the LE preimage is computed.

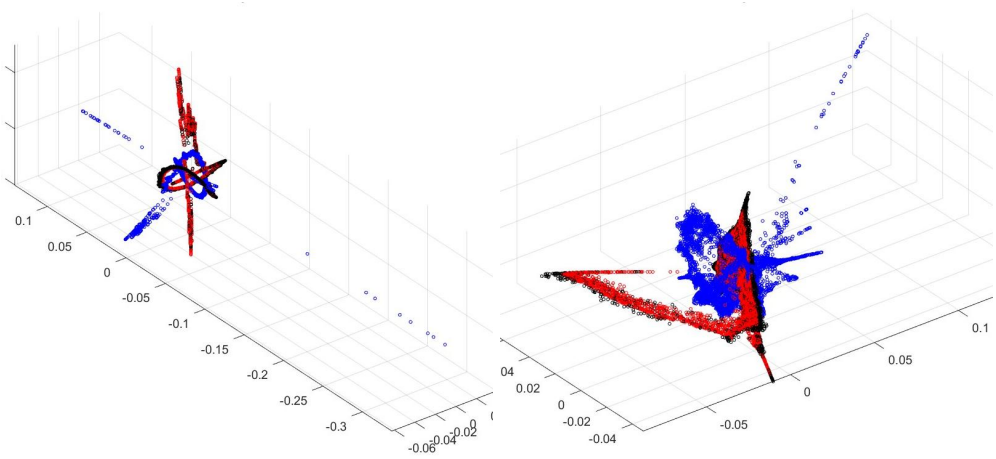
The T1w and T2w embeddings are depicted in Figure 7.5, with low resolution T1w pixels shown in blue, low resolution T2w pixels in black, and high resolution T2w pixels in red. Figure 7.5d show that without spatial information, the embeddings align in curve-like shapes, similar to the equivalent situation shown in Figure 6.5. Further, the two embeddings in Figure 7.5d both have portions which align to an axis. The observation that these points have coordinates close to 0 in a few of the first eigenvectors usually indicates a graph which is not very well-connected. This notion is supported by literature which connects the level of connectedness between clusters and the support of Laplacian eigenvectors [27]. Once spatial data is incorporated in Figure 7.5e, we see that the points of both T1w and T2w embeddings are more varied in all three depicted dimensions.

Finally, the preimage results in Figure 7.6 exhibit a large improvement through the addition of spatial data. The spatial preimage has significantly less incorrect texture than its counterpart without location data. We attribute this to a weakly connected graph in the spectral-only case; many loosely connected parts of the graph could be permuted during the preimage process without spatial regulation. We also saw a similar increase in the color image MMSR problem in Table 6.1. Therefore, we choose to incorporate spatial location of pixels during graph construction in

Figure 7.5: *Adding spatial information during graph formation for BraTS Volume 1, Slice 68. The low resolution images are downsized by a scaling factor of $r = 2$ in both spatial dimensions. Points in red represent pixels from the low resolution T2w image, points in black represent the pixels from the high resolution T2w image, and points in blue represent pixels from the low resolution T1w image. Note that in (d), there are T1w points, shown in blue, which align parallel to an axis. We take this to mean that the T1w graph is not connected to some degree, as there are points which have coordinates of 0 across eigenvectors for smaller eigenvalues. The two embeddings are depicted at different scales to highlight the geometry of blue points in (d).*

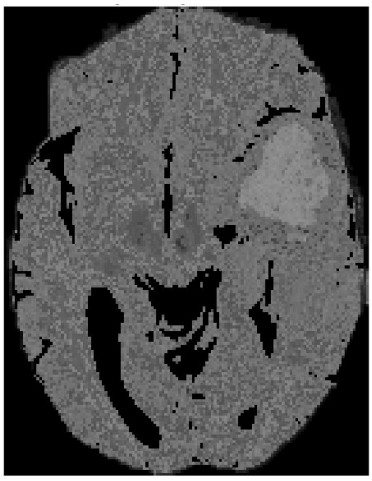


(a) *T1w Low* **(b)** *T2w High* **(c)** *T1w High*

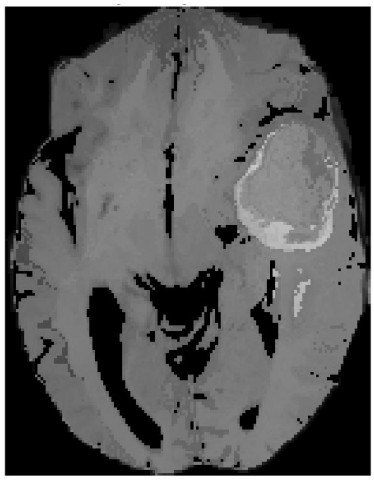


(d) *Embedding using only spectral data* **(e)** *Embedding using spectral and spatial data*

Figure 7.6: *Results of adding spatial information in graph formation for BraTS Volume 1, Slice 68. Note that the addition of spatial data in (b) has better metrics and results in an image with less heterogeneity when compared to (a) which does not have spatial information.*



(a) *Spectral-only data, T1w*



(b) *Spectral and spatial data, T1w*

	Spectral-Only Data	Spatial and Spectral Data
PSNR	22.07	**27.37**
SSIM	0.628	**0.852**

our method.

7.3 Comparison of Graph Matching and Clustering in Preimages

In Section 4.2, we present two methods for mapping between LE embeddings spaces for data of different modalities: clustered rotation and graph matching. The clustered rotations of Section 4.2.2 map between spaces by performing a clustering on an embedding, then using a Procrustes rotation into the target space for each cluster, while the graph matching approach of Section 4.2.3 exploits the known correspondence between pixels of the low resolution T1w and T2w images to map high resolution points into the target space. Here, we will examine the merits of both options in our MRI MMSR problem. Using BraTS volume 1, slice 68 as the test image, we construct a high resolution T1w image from a combination of a high resolution T2w image

and a low resolution T1w image through k-means clustered rotation, curvature-based clustered rotation, and graph matching. Both clustering methods use 3 clusters, chosen due to the geometric features in the T2w LE embedding of Figure 7.7a and the different shades visible in Figure 7.2a.

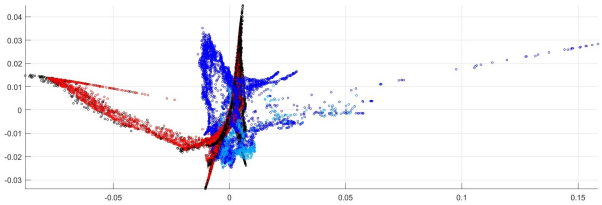
The 4 plots of Figure 7.7 are all projections of 40-dimensional LE embeddings on to the plane of the first and third eigenvectors. The T1w LE embedding, depicted in blue in Figure 7.7a, features a mass of points toward the left side of the plot and a trail of points leaking out to the top right corner. All 3 mappings into the T1w embedding space do a good job of spreading points out on the line, but graph matching provides a much better shape fitting everywhere else. For example, the k-means clustered rotation sends T2w points, shown in a red-to-yellow gradient, far away from T1w points in the blue gradient.

The preimage results make the difference between the methods even more stark. The k-means preimage in Figure 7.8a has very poor performance on the tumor located in the top right of the image, as it depicts high variation in that region which is not present in either the T2w image or the T1w reference image. The gap in PSNR and SSIM between graph matching and the clustered rotation methods is in agreement with the experiments on color MMSR in Table 6.1. Thus, we proceed with graph matching as our preferred mapping between LE embedding spaces moving forward.

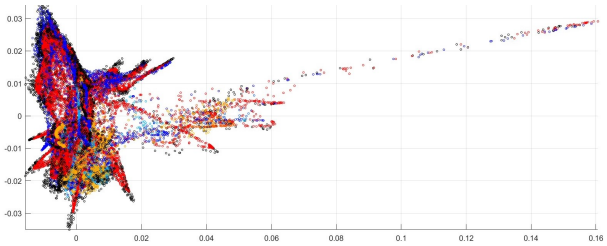
7.4 Joint Modality Inputs

In Section 4.4.1.2, we present an option in creating graphs from pixel data which uses a combination of two images, inspired by the works of Lu et al. [83] and Dong et al. [45]. Here, we

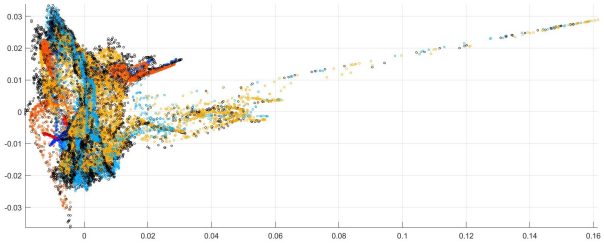
Figure 7.7: *Comparison of LE embeddings for clustering and matching approaches for BraTS1 Slice 68. With the eigenvectors placed in order according to the size of their eigenvalues, from smallest to greatest, each plot shows the* 1*st eigenvector coordinate in the x-axis and the* 3*rd eigenvector coordinate in the y-axis. In (a), the T1w points are given in blue, the low resolution T2w points in black, and the high resolution T2w points in red. For the other plots, we can gauge quality of mapping between embedding spaces by inspecting how close the shape of the plot is to the shape of the blue points in (a). The matching plot (d) appears to follow the correct shape better than the other two plots.*



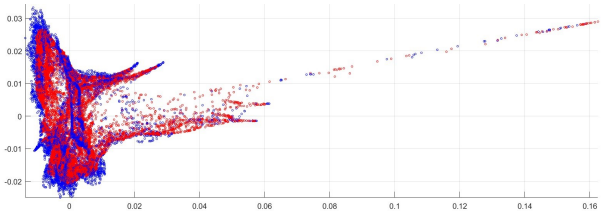
(a) *T1w (blue) and T2w (black + red out-of-sample) embeddings*



(b) *After k-means clustering and rotation*



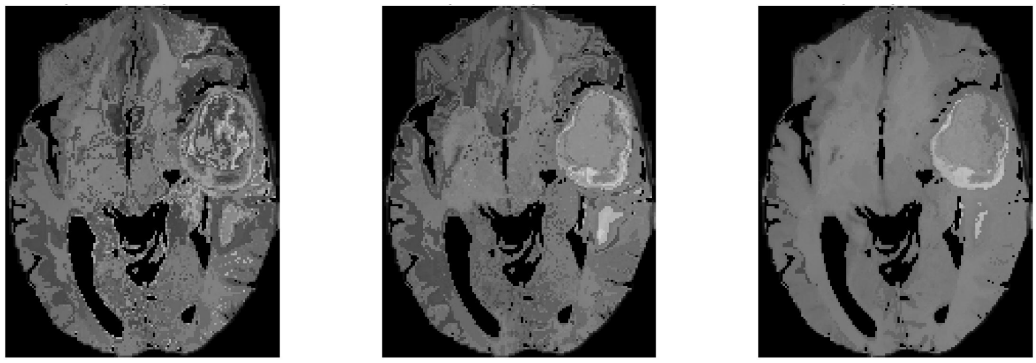
(c) *After curvature-based clustering and rotation*



(d) *After matching*

Figure 7.8: *Preimage Results for Clustering and Matching, BraTS1 Slice 68. The preimage obtained via graph matching (c) has the best error metrics and appears smoother than the other two. In addition, the preimage obtained using the curvature-based clustering (b) appears not to vary as much as the k-means preimage (a) over small spatial pixel neighborhoods.*

(a) *k-means* **(b)** *curvature* **(c)** *matching*

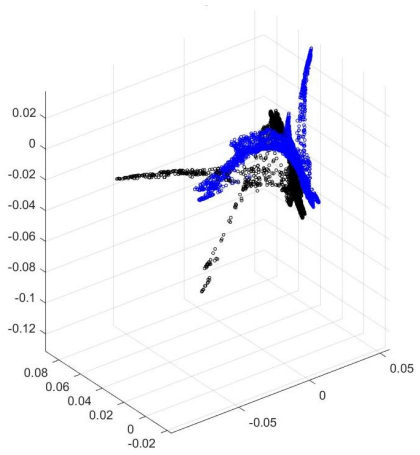


	k-means	curvature	matching
PSNR	20.31	22.14	**27.37**
SSIM	0.600	0.672	**0.852**

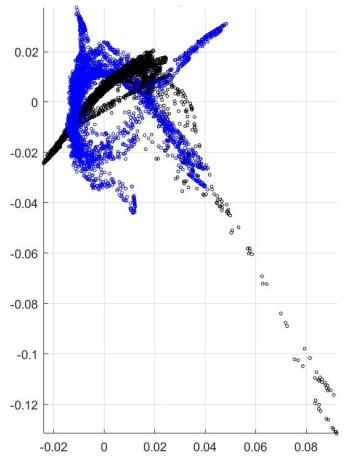
test the method in our MMSR problem. Our method uses LE preimages of the pixels of the high resolution T2w image from the T1w embedding space. These preimages are determined by the location of the points in the T1w embedding space. Thus, it is crucial that the two embeddings, created from the low resolution MR images, have similar shapes.

Let us consider the properties of the T1w and T2w images in order to create good shape matching. The T2w images of the same slices in the BraTS data set present much higher dynamic range than their T1w counterparts. This is due to a variety of factors, including the nature of the brain tissue [63], the pulse parameters used by clinicians in creating the images, and any processing steps which are applied to the images [88]. As a result, we expect that the mean distance of pixel vectors from T2w images is higher than those of their corresponding T1w images. This creates a disparity in the embeddings over the two modalities.

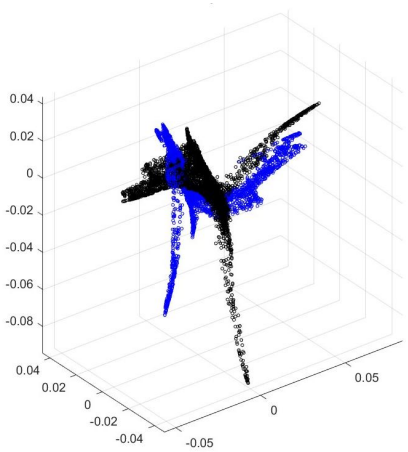
Figure 7.9: *Comparison of LE embeddings with single and joint image inputs, BraTS27 Slice 63. Shown are the T1w and T2w LE embeddings using a single image input (a) - (b) compared to the same embeddings using a joint image embedding (c) - (d), with T1w points in blue and T2w points in black. We highlight the difference in scale of the vertical axes in (b) and (d). The joint image T2w embedding results in a better size match to the T1w embedding.*



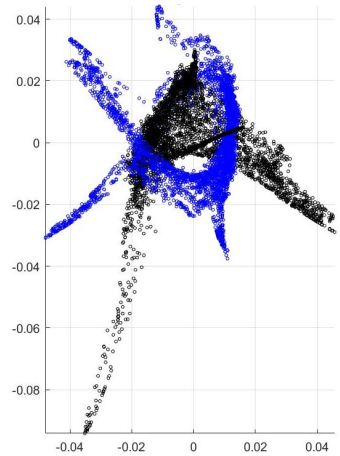
(a) *Single image input, T1w LE in blue, T2w LE in black*



(b) *Single image input, Projection onto 2nd and 3rd eigenvectors*



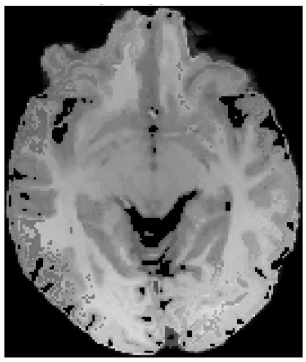
(c) *Joint image input, T1w LE in blue, T2w LE in black*



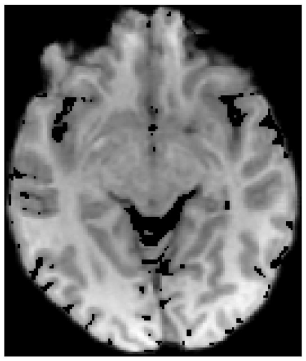
(d) *Joint image input, Projection onto 2nd and 3rd eigenvectors*

The Laplacian eigenmap embedding is designed to represent the local geometry on a data set, respecting small distance between connected pixel vectors. The distance between unconnected pixel vectors in LE embedding space then is comparable to distances on the graph which generates the embedding [32]. Long distances between unconnected pixel vectors then require a large scale to encode the path between their embedded points. An example of this phenomenon can be seen in Figures 7.9a and 7.9b. These depict embedding spaces for BraTS volume 27, slice 63. Here, we show the two embeddings of low resolution T1w and T2w pixels, superimposed together on one set of axes. We especially note that the T2w pixels, shown in black, are elongated in the direction of the 3rd eigenvector.

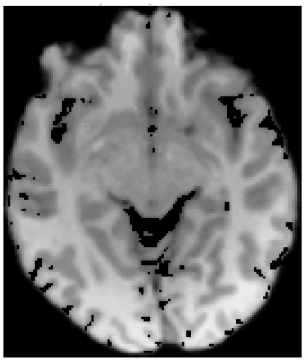
Figure 7.10: *Preimage results for a BraTS with single and joint image inputs, BraTS27 Slice 63. The joint image input case (c) results in a much more accurate T1w image than the single image input case (a), particularly in the bottom quarter of the image.*



(a) *T1w preimage, single image input*



(b) *T1w reference image*



(c) *T1w Preimage, joint image input*

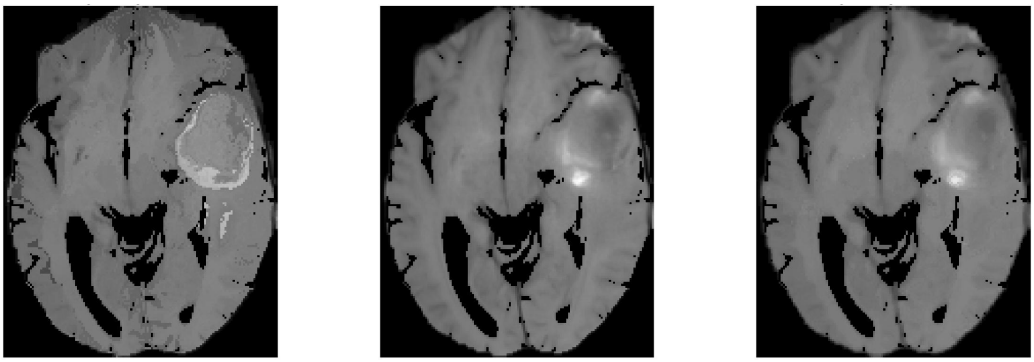
	Single Image	Joint Image
PSNR	21.53	**34.32**
SSIM	0.756	**0.970**

(d) *PSNR and SSIM for the two preimages*

To work around this issue, we use the joint input image method described in Section 4.4.1.2. The low resolution T1w image is upscaled using a bicubic interpolation implemented in MATLAB using the `resize` built-in function. These interpolated T1w pixels are then concatenated

on to the original high resolution T2w pixel vector. Similarly, the low resolution T2w pixels which generate the T2w LE embedding have the low resolution T1w pixels adjoined to their pixel vectors. Now, we show the same embeddings in Figures 7.9c and 7.9a, where the T2 embedding was created using the method of Section 4.4.1.2. We can see that the elongation from the previous embedding is reduced, resulting in two embeddings which more closely match in shape and scale.

Figure 7.11: *Preimage results for a BraTS with single and joint image inputs, BraTS1 Slice 68. While the recovered T1w image using joint image input (c) is clearly a better quality reconstruction than in the single image input case (a), there are some features the preimage fails to capture. For example, the tumor present in the center-right of the reference image (b) is brighter than that of the preimage (c).*



(a) *Recovered T1w using single image input*

(b) *High resolution T1w reference image*

(c) *Recovered T1w using joint image input*

	Single Image	Joint Image
PSNR	27.37	**34.73**
SSIM	0.852	**0.962**

(d) *PSNR and SSIM for the two preimages*

We also include the results of the same analysis on BraTS volume 1, slice 68 in Figure 7.11 for comparison to previous results in this chapter. Again, we see a dramatic increase in both PSNR and SSIM through this technique. In particular, it is very difficult to distinguish the visual difference between the preimage from the joint image method in Figure 7.11c and the high resolution reference T1w image in Figure 7.11b. One noticeable distinction is that the preimage

is not as bright as the reference image at the tumor, the bright spot located in the center-right of the image.

Table 7.2: *Mean PSNR and SSIM for T1w recovery values for 4 brain volumes, comparing single and joint image inputs. The different columns consider low resolution images which are downsized at scaling factors $r = 2, 3$ and 8.*

	$r = 2$	$r = 3$	$r = 8$	Comments
BrainWeb Normal Single Input	25.86	27.04	25.95	From T2w to T1w – Mean PSNR
	0.844	0.903	0.906	From T2w to T1w – Mean SSI
BrainWeb Normal Joint Input	**33.99**	**38.11**	**28.22**	From T2w to T1w – Mean PSNR
	0.944	**0.979**	**0.916**	From T2w to T1w – Mean SSI
BraTS001 Single Input	27.47	26.68	24.28	From T2w to T1w – Mean PSNR
	0.903	0.902	0.884	From T2w to T1w – Mean SSI
BraTS001 Joint Input	**33.75**	**30.49**	**24.66**	From T2w to T1w – Mean PSNR
	0.967	**0.958**	**0.900**	From T2w to T1w – Mean SSI
BraTS027 Single Input	20.70	20.31	18.68	From T2w to T1w – Mean PSNR
	0.744	0.728	0.702	From T2w to T1w – Mean SSI
BraTS027 Joint Input	**34.46**	**30.40**	**21.11**	From T2w to T1w – Mean PSNR
	0.968	**0.936**	**0.729**	From T2w to T1w – Mean SSI
BraTS164 Single Input	24.77	24.63	**22.51**	From T2w to T1w – Mean PSNR
	0.858	0.864	**0.852**	From T2w to T1w – Mean SSI
BraTS164 Joint Input	**29.38**	**26.69**	21.84	From T2w to T1w – Mean PSNR
	0.938	**0.902**	0.791	From T2w to T1w – Mean SSI

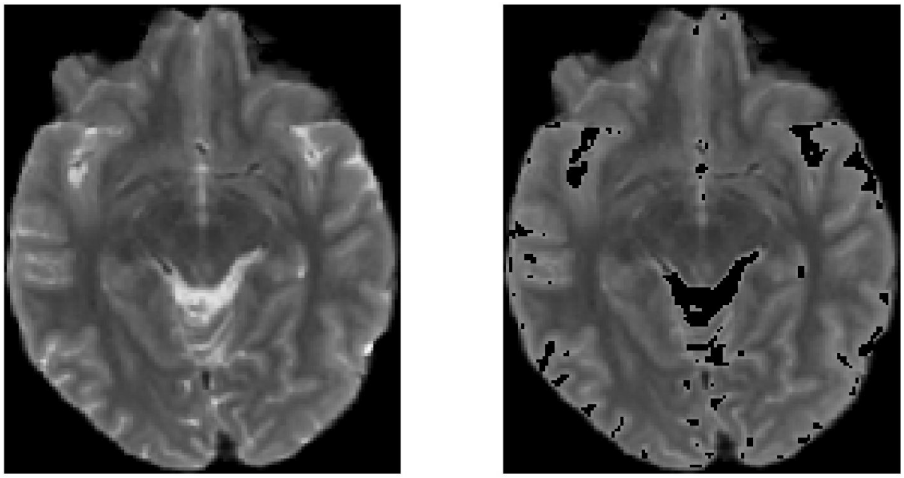
Finally, we present a table of results comparing the joint image input method to using only a single input. This study is conducted using 11 axial slices from the BrainWeb normal volume, 10 axial slices from BraTS volume 1, 13 axial slices from BraTS volume 27, and 11 axial slices from BraTS volume 164. These slices are chosen as they exhibit distinct image structures, both in the topography of the brain images and in the intensity values attained. We perform the MMSR using 3 different scaling factors for creating the low resolution image, $r = 2, 3$ and 8. The per-volume means of PSNR and SSIM are then calculated. We see that in all but one case, the joint image option provides a clear improvement in both PSNR and SSIM. The one outlier is for the BraTS 164 volumes with scaling factor $r = 8$, in which the images have a resolution of 30×30. Because of this, we believe that this outlier is not significant because it is not a realistic representation of

an actual low resolution MRI. Because the joint image input method provides superior results, we implement it in our MMSR method.

7.4.1 Effect of Removing CSF in MR Images

We use this section as a brief aside to justify our decision to edit the MR images by removing cerebrospinal fluid (CSF), and to explore the effect of an augmentation of the Nyström out-of-sample extension presented in Section 3.5. This provides the last part of our overall MRI superresolution pipeline.

Figure 7.12: *A BraTS T2w image with and without CSF, BraTS27 Slice 63. The CSF is removed by thresholding intensity values above a certain point, followed by manual correction using Slicer3D software.*



(a) *T2w image with CSF* **(b)** *T2w image with CSF removed*

We remove CSF for at least two potential reasons. The first is because it isn't part of the brain *parenchyma*, the tissue that makes up the connective and functional part of the brain. A reasonable rebuttal to that point is that the relationship we seek, between T1w and T2w intensity values, is just as valid on CSF as it is on more solid tissue like gray and white matter. The second reason is that we see from the previous section that having a high dynamic range may not be

beneficial for creating a T1w image.

We remove CSF via a simple intensity thresholding applied to the T2w image. CSF presents as bright on T2w images and can be found around the boundary of the brain and in the ventricles in the center. Then we can just choose to remove all pixels above a certain value, determined by visual inspection per brain volume. By removing pixels at the corresponding pixel locations in the T1w image, we can also mask out the CSF there. Once we have this for a high resolution T2w image, downsizing provides a CSF mask that can be used on the low resolution images as well. Of course, this method works because we have access to high resolution versions of images of both modalities. In the more *realistic* case of high resolution in only one modality, T1w in the less favorable case, we can obtain a slightly less accurate CSF removal using an upsizing of the T2w image.

The three CSF masking options we examined were masking out the CSF *before* any calculations are done, masking out CSF *after* the construction of the preimage, and masking out CSF in the middle of the process, right before the out-of-sample extension. We will refer to these 3 CSF removal orders as *masking first*, *masking last*, and *masking in the LE embedding*, respectively. Figure 7.13 shows the order of CSF removal in each of the three methods. CSF removal is then taken to be a regular step in our method instead of an optional optimization.

Masking in the LE means that we use CSF pixels in low resolution images in order to create the embeddings, but the out-of-sample extension is built from linear combinations of embeddings of non-CSF pixels. Recall from Section 3.5 that we have the extension $\widetilde{\Phi}(y)$ as a sum over $\mathcal{N}_y$, the set of neighbors in the sample set X to some out-of-sample point y. If we instead restrict the neighborhood to be inside a subset $X_0 \subseteq X$, then we still obtain a map $\widetilde{\Phi}_0$ into the LE embedding space of X. In this case, we take X_0 to be the set of pixel vectors which do not represent CSF.

Figure 7.13: *A diagram depicting the three options of CSF removal. Masking first refers to removing CSF prior to any other calculations. Masking in LE embedding means that CSF pixels are used to create the embeddings, but then are not carried through the rest of the algorithm. Masking last refers to only removing CSF after the algorithm is completed, right before we measure the error of the new high resolution image.*

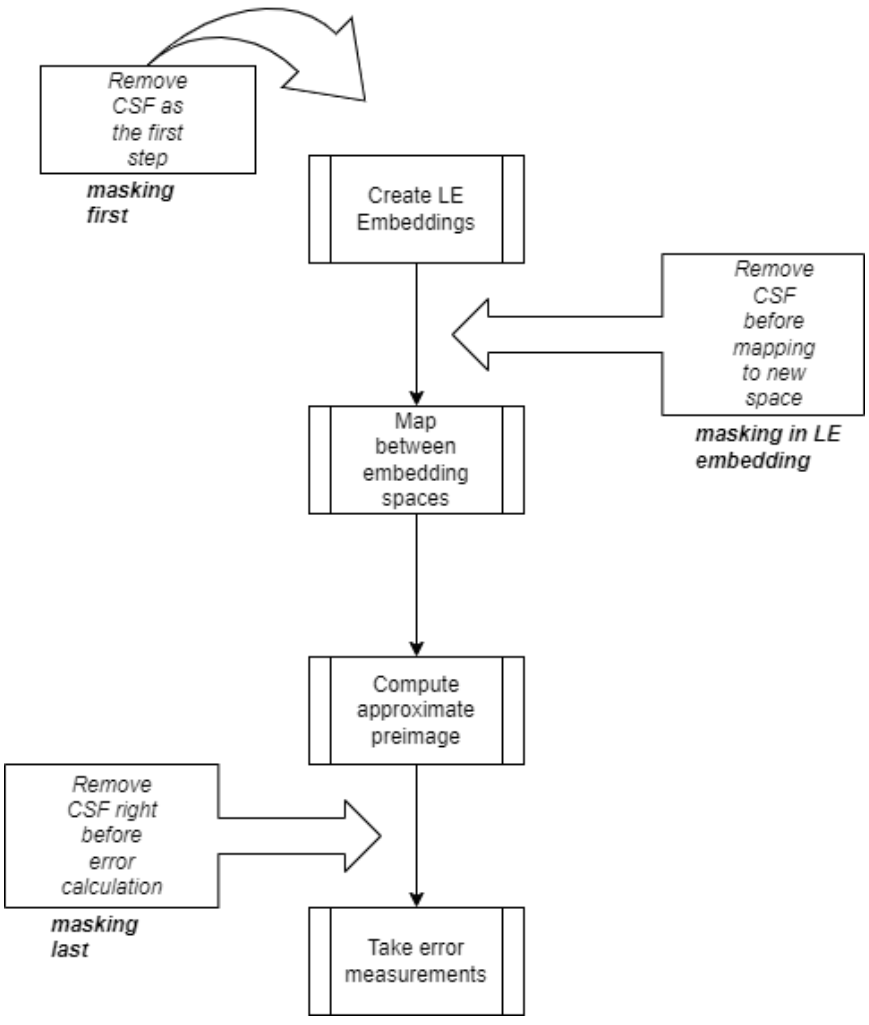
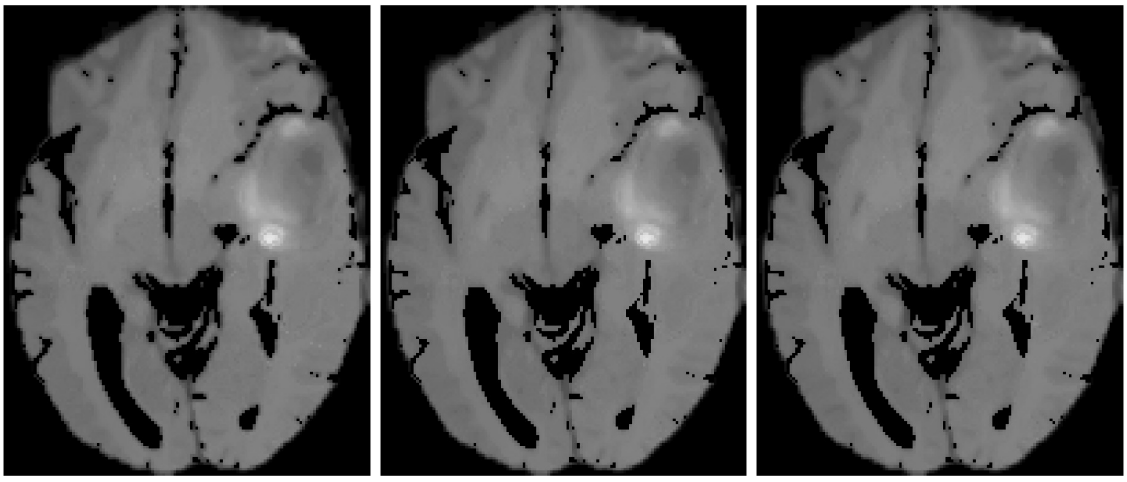


Figure 7.14: *Comparing CSF removal orders for BraTS volume 1, slice 68. From left to right we have the preimages from masking first (a), masking in the LE embedding (b), and masking last (c). The T1w preimages are shown, using the joint image input method. Although we get good results in all 3 cases, masking first has a noticeable disadvantage in PSNR and SSIM, shown in the table (d) below.*

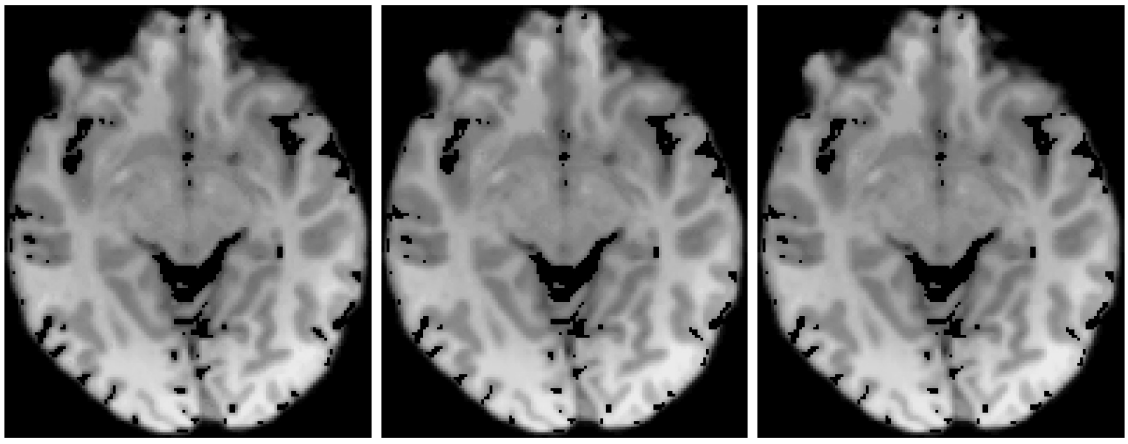


(a) *T1w, CSF removed first* **(b)** *T1w, CSF removed in LE* **(c)** *T1w, CSF removed last*

	Remove First	Remove Middle	Remove Last
PSNR	31.33	**34.79**	34.73
SSIM	0.948	0.9620	**0.9623**

(d) *Table of PSNR and SSIM values*

Figure 7.15: *Comparing CSF removal orders for BraTS volume 27 slice 63. From left to right we have the preimages from masking first (a), masking in the LE embedding (b), and masking last (c). The T1w preimages are shown, using the joint image input method. While both have better PSNR and SSIM, shown in (d), than masking first, the two later masking orders have almost indistinguishable results.*

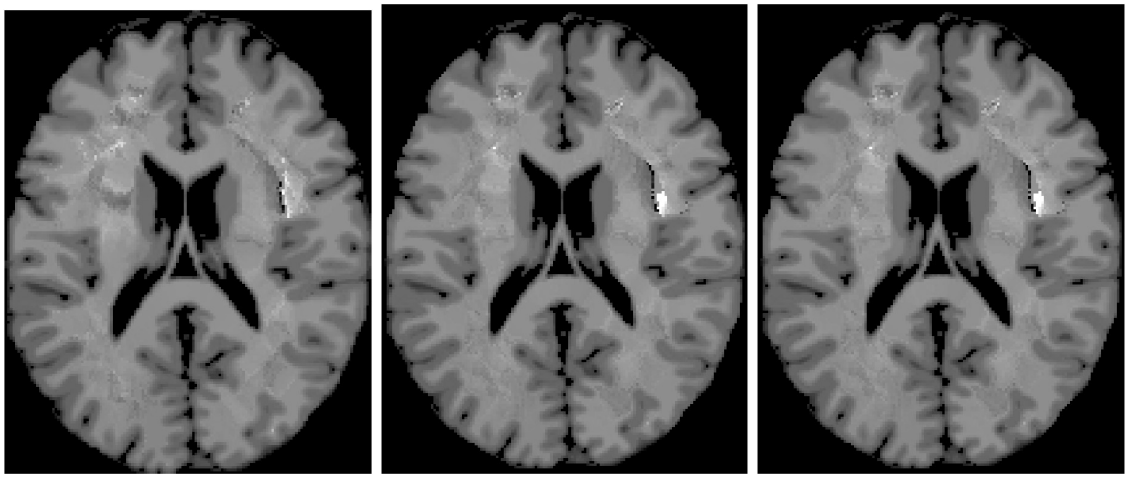


(a) *T1w, CSF removed first* **(b)** *T1w, CSF removed in LE* **(c)** *T1w, CSF removed last*

	Remove First	Remove Middle	Remove Last
PSNR	30.90	34.24	**34.32**
SSIM	0.957	0.9700	**0.9704**

(d) *Table of PSNR and SSIM values*

Figure 7.16: *Comparing CSF removal orders for BrainWeb normal volume, slice 91.*



(a) *T1w preimage, CSF removed first*

(b) *T1w preimage, CSF removed first*

(c) *T1w preimage, CSF removed last*

	Remove First	Remove Middle	Remove Last
PSNR	**30.57**	30.37	30.39
SSIM	0.916	0.926	**0.927**

(d) *Table of PSNR and SSIM values*

We tested the effect of the three different mask orders in Figures 7.14, 7.15, and 7.16, which feature a slice from BraTS volumes 1 and 27 and the BrainWeb normal volume, respectively. Our original was that the most straightforward option, removing the CSF before any calculations are done, would provide the best results. The data shows that this is certainly not the case. We attribute this to the idea that having the CSF pixels included in graph creation still provides useful information to improve the mapping between embedding spaces. Thus removing CSF pixels before the embeddings are created reduces the information content used in creating the mapping. While we can conclude that masking first does not outperform the other two methods, we require more criteria to determine which, if any, of the two remaining mask orders is superior. To do this, we repeat this preimage comparison for the 45 slices over 4 volumes described in Section 7.4.

Table 7.3: *Two tailed, paired t-test results comparing the means of the PSNR and SSIM values for two different CSF masking orders. The full sample has 45 images, 34 from the BraTS set and 11 from BrainWeb. Samples which provide statistically significantly different means are denoted in* ***bold****. After Bonferroni correction, many of the differences in means are statistically insignificant.*

2×2	Mask Middle Mean	Mask Last Mean	p-value	
All Slices	32.91	32.95	$1.7 \cdot 10^{-1}$	PSNR
	0.954	0.954	$2.4 \cdot 10^{-1}$	SSIM
Normal BrainWeb	**33.80**	**33.99**	$\mathbf{1.9 \cdot 10^{-4}}$	**PSNR**
	0.943	**0.944**	$\mathbf{1.6 \cdot 10^{-5}}$	**SSIM**
BraTS Volumes 1, 27, 164	32.62	32.60	$7.1 \cdot 10^{-1}$	PSNR
	0.958	0.958	$6.6 \cdot 10^{-1}$	SSIM
3×3	Mask Middle Mean	Mask Last Mean	p-value	
All Slices	31.36	31.40	$3.0 \cdot 10^{-2}$	PSNR
	0.943	0.943	$5.4 \cdot 10^{-2}$	SSIM
Normal BrainWeb	37.98	38.11	$5.1 \cdot 10^{-3}$	PSNR
	0.9790	**0.9793**	$\mathbf{3.0 \cdot 10^{-3}}$	**SSIM**
BraTS Volumes 1, 27, 164	29.22	29.23	$5.1 \cdot 10^{-1}$	PSNR
	0.931	0.932	$2.1 \cdot 10^{-1}$	SSIM
8×8	Mask Middle Mean	Mask Last Mean	p-value	
All Slices	**23.97**	**23.82**	$\mathbf{1.8 \cdot 10^{-4}}$	**PSNR**
	0.828	0.823	$8.1 \cdot 10^{-1}$	SSIM
Normal BrainWeb	28.42	28.23	$3.7 \cdot 10^{-2}$	PSNR
	0.913	0.916	$1.6 \cdot 10^{-2}$	SSIM
BraTS Volumes 1, 27, 164	22.53	22.39	$2.4 \cdot 10^{-3}$	PSNR
	0.801	0.799	$6.2 \cdot 10^{-2}$	SSIM

We perform hypothesis testing using t-tests to seek out a statistically significant difference in the performance of the two methods, masking in the LE and masking last. We consider 3 sample sets, all from the set of 45 image slices described in Section 7.4: the full set of the selected 45 slices among both BraTS and BrainWeb data set, the 11 BrainWeb slices of that set, and the 34 remaining BraTS slices. Our null hypotheses are that after application of either masking in the LE or masking last, the resulting PSNR or SSIM values are from the same empirical distribution. As we are conducting multiple tests, we utilize the Bonferroni correction [47] in order to decrease the likelihood of a false positive result. For the set of 45, we set our significance level to $\mu_1 = \dfrac{0.05}{45} = 1.\overline{11} \cdot 10^{-3}$, and for the other two sets of 34 and 11, the significance levels are $\mu_2 = \dfrac{0.05}{34} \approx 1.471 \cdot 10^{-3}$ and $\mu_3 = \dfrac{0.05}{11} = 4.\overline{54} \cdot 10^{-3}$, respectively.

The results of the hypothesis testing are given in Table 7.3. As evident from the results, the difference between the performance of these two CSF removal orders are subtle and highly dependent on the specific experiment. In addition, the results of the t-tests differ between the two comparison metrics. With these factors at hand, an empirical basis for choosing one order over the other remains elusive. Therefore, we proceed by removing CSF in the LE for computational efficiency. In this scheme, the out-of-sample extension of each high resolution T2w pixel invokes a nearest neighbor search in the smaller subset $X_0 \subseteq X$ composed of non-CSF pixels. This results in slightly fewer distance calculations per pixel.

7.5 Results

In this section, we test the capabilities of our multimodal superresolution method applied to MRI, using the choices described previously in this chapter. In summary, we begin with a

high resolution source image and a low resolution target image. We create pixel vectors for these images. The T1w pixel vectors use intensity and spatial location data. The T2w pixel vectors, both high and low resolution version, use intensity and spatial data as well as the intensity information of their corresponding T1w pixel. In the case of high resolution T2w pixels, the T1w information comes from interpolating the low resolution T1w image. Once the pixel vectors for all three images are formed, we then create target and source LE embeddings using the low resolution pixel vectors of both modalities. The LE embedding is formed by creating a graph Laplacian with the pixel vectors as graph nodes, then solving for the eigenvectors of that graph Laplacian.

Since the two low resolution images have the same dimensions, their pixels can be identified. This provides the seeds for a mapping from the source LE space to the target LE space. The mapping is formed by first sending low resolution source embedded points to their corresponding low resolution target embedded points. From here, we use an adapted Nyström out-of-sample extension to embed the high resolution source pixels into the target space. Each such high resolution pixel is mapped based on its nearest low resolution source neighbors. As a result, the high resolution source pixels are embedded into the target space, retaining local similarity information from both modalities.

Finally, an LE preimage is used on the high resolution source pixels out of the target space. These preimage values are the intensity values which form a high resolution approximate target image.

Figures 7.17 through 7.19 give a direct comparison of our method and bicubic interpolation on 4 different slices from 4 different volumes. Table 7.4 presents PSNR and SSIM means, comparing the two methods over 45 slices in 4 volumes. The general trend in the data is that

Algorithm 7.1 MRI multimodal superresolution using Laplacian Eigenmaps

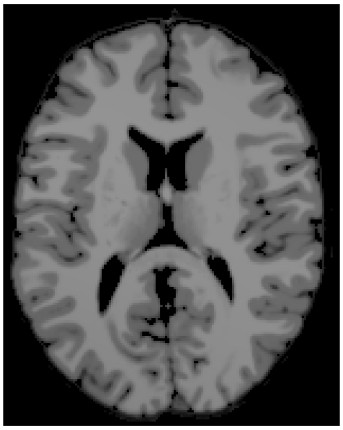
1: Create data vectors of low resolution T1w image using intensity and spatial data.
2: Perform bicubic interpolation on T1w image.
3: Create data vectors from T2w pixels using intensity, spatial data, and intensity from interpolated T1w image.
4: Create T1w and T2w Laplacian eigenmaps embedding spaces from low resolution pixels.
5: Match the non-CSF embedded T2w pixels to non-CSF embedded T1w pixels by spatial coordinates.
6: Perform an out-of-sample extension to map high resolution T2w pixels into T2w LE embedding space, based on proximity of data vectors to those of low resolution, non-CSF T2w pixels.
7: Use the identification of low resolution T1w and T2w non-CSF pixels to map high resolution T2w pixels into T1w embedding space.
8: Compute the LE preimage of the high resolution, non-CSF T2w pixels from the T1w embedding space, forming a high resolution approximate T1w image.

Table 7.4: *A comparison of superresolution results on a T1w MRI using our MMSR method vs. bicubic interpolation. The larger PSNR or SSIM value between the two methods is denoted in* ***bold****. The means are taken over* 11 *normal BrainWeb slices and* 34 *BraTS slices, spread over* 3 *volumes. The different columns consider low resolution images which are downsized at scaling factors* $r = 2, 3$ *and* 8*. While interpolation provides better results for scaling factor* $r = 2$*, the data fusion method out-performs interpolation when using downsized images with a scaling factor of* $r = 3$ *or more.*

		$r = 2$	$r = 3$	$r = 8$	Comments
Normal BrainWeb	LE Preimage	33.80	**37.98**	**28.41**	– Mean PSNR
		0.943	**0.979**	**0.913**	– Mean SSI
	Bicubic Interpolation of T1w	**37.75**	33.31	24.84	– Mean PSNR
		0.981	0.949	0.766	– Mean SSI
BraTS Volume 1	LE Preimage	**33.80**	**30.38**	**24.57**	– Mean PSNR
		0.967	**0.9571**	**0.900**	– Mean SSI
	Bicubic Interpolation of T1w	32.27	28.51	21.62	– Mean PSNR
		0.972	0.937	0.792	– Mean SSI
BraTS Volume 27	LE Preimage	34.32	**30.38**	**21.21**	– Mean PSNR
		0.967	**0.936**	**0.728**	– Mean SSI
	Bicubic Interpolation of T1w	**34.82**	29.94	19.80	– Mean PSNR
		0.976	0.926	0.623	– Mean SSI
BraTS Volume 164	LE Preimage	**29.54**	**26.78**	**22.26**	– Mean PSNR
		0.939	**0.903**	**0.796**	– Mean SSI
	Bicubic Interpolation of T1w	27.43	24.18	18.77	– Mean PSNR
		0.926	0.850	0.618	– Mean SSI

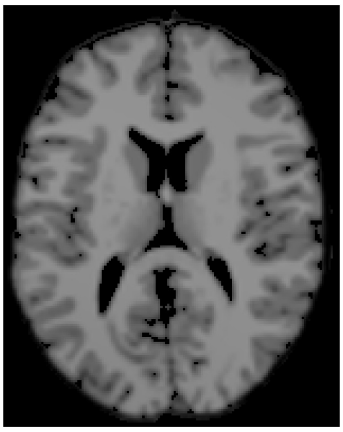
our method performs about equally or worse than interpolation in the case of downsizing by a scaling factor of $r = 2$. We attribute this to the fact that interpolation exclusively uses the spatial neighbors of pixels to determine new values. Our method, in contrast, takes spatial and intensity closeness into account. For downsizing on a small scale, relying on non-spatial factors likely causes more local variation of intensity than desired. However, at increased levels of deterioration of the image, our method overtakes interpolation in both metrics. Further, our method maintains, or even exceeds, its level of performance at larger scales. For example, Figure 7.17 shows that our method is able to recover a better quality image with an $r = 3$ downsizing than interpolation can with an $r = 2$ downsizing. We note that since this example is from the simulated BrainWeb data set, it may not reflect the performance of our method on actual MRI. Nevertheless, we believe that our results on BraTS images demonstrate the capability of our data fusion model to solve this problem by using data in one modality to compensate for a lack in another modality.

Figure 7.17: *Comparison of our MMSR method and bicubic interpolation on an axial slice, normal BrainWeb volume, slice 86. The high resolution T1w bicubic interpolations with scaling factors $r = 2$, $r = 3$, and $r = 8$ are shown in (c), (d), and (e), respectively. The high resolution T1w LE preimages through our method using low resolution T1w images with scaling factors of $r = 2$, $r = 3$, and $r = 8$ are shown in (f), (g), and (h), respectively. In the table (b) below, "BI" means bicubic interpolation and "LE" means LE preimage. The larger PSNR or SSIM value between the two methods is denoted in **bold**.*

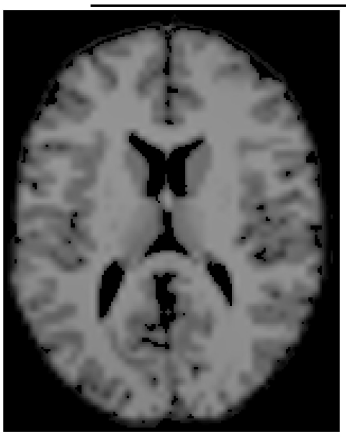


(a) *T1w Reference Image*

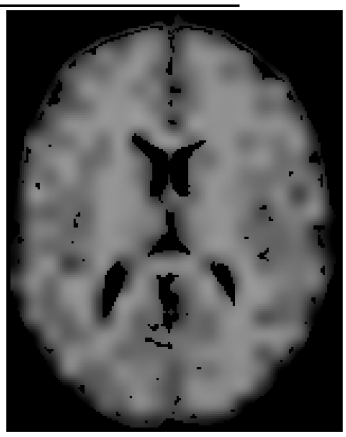
		$r = 2$	$r = 3$	$r = 8$
BI	PSNR	**37.04**	32.70	24.73
	SSIM	**0.976**	0.939	0.750
LE	PSNR	36.69	**38.71**	**29.58**
	SSIM	0.965	**0.983**	**0.914**



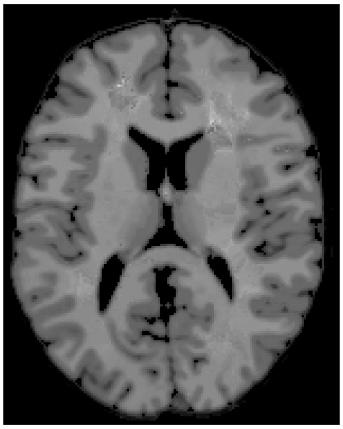
(c) $r = 2$ *Interpolation*



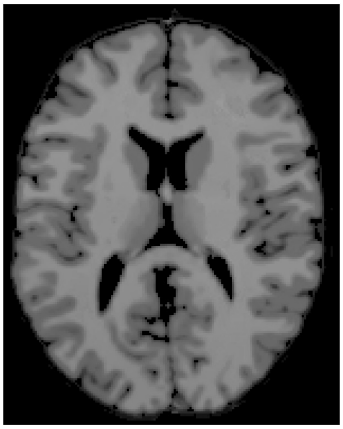
(d) $r = 3$ *Interpolation*



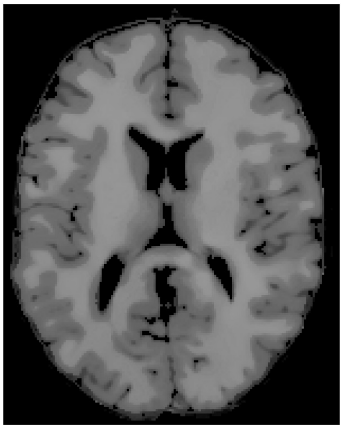
(e) $r = 8$ *Interpolation*



(f) $r = 2$ *LE MMSR*

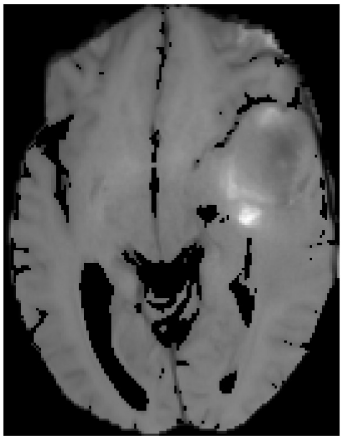


(g) $r = 3$ *LE MMSR*



(h) $r = 8$ *LE MMSR*

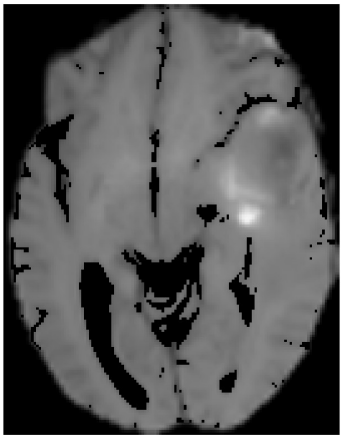
Figure 7.18: *Comparison of our MMSR method and bicubic interpolation on an axial slice, BraTS volume 1, slice 68. The high resolution T1w bicubic interpolations with scaling factors $r = 2$, $r = 3$, and $r = 8$ are shown in (c), (d), and (e), respectively. The high resolution T1w LE preimages through our method using low resolution T1w images with scaling factors of $r = 2$, $r = 3$, and $r = 8$ are shown in (f), (g), and (h), respectively. In the table (b) below, "BI" means bicubic interpolation and "LE" means LE preimage. The larger PSNR or SSIM value between the two methods is denoted in* ***bold****.*



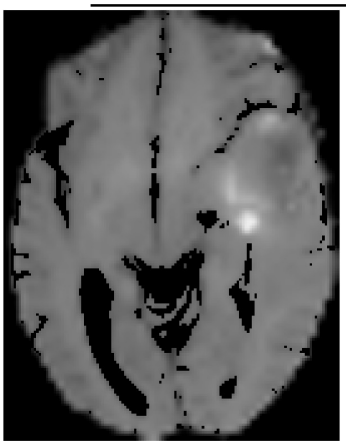
(a) *T1w Reference Image*

		$r = 2$	$r = 3$	$r = 8$
BI	PSNR	34.20	30.95	23.94
	SSIM	**0.975**	0.943	0.775
LE	PSNR	**34.79**	**32.14**	**26.72**
	SSIM	0.962	**0.951**	**0.888**

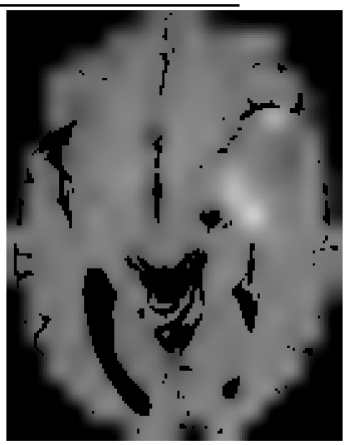
(b)



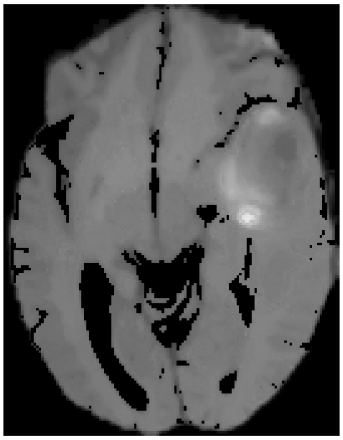
(c) $r = 2$ *Interpolation*



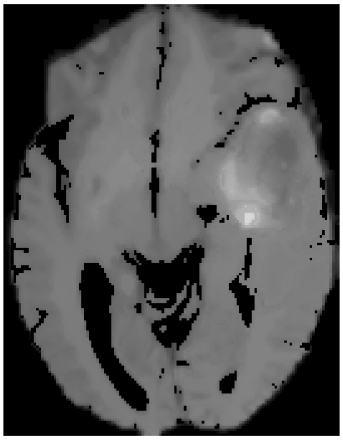
(d) $r = 3$ *Interpolation*



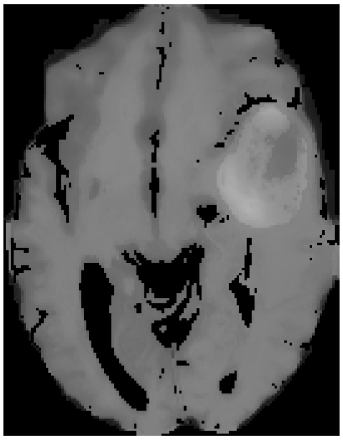
(e) $r = 8$ *Interpolation*



(f) $r = 2$ *LE MMSR*



(g) $r = 3$ *LE MMSR*



(h) $r = 8$ *LE MMSR*

Figure 7.19: *Comparison of our MMSR method and bicubic interpolation on an axial slice, BraTS volume 164, slice 88. The high resolution T1w bicubic interpolations with scaling factors $r = 2$, $r = 3$, and $r = 8$ are shown in (c), (d), and (e), respectively. The high resolution T1w LE preimages through our method using low resolution T1w images with scaling factors of $r = 2$, $r = 3$, and $r = 8$ are shown in (f), (g), and (h), respectively. In the table (b) below, "BI" means bicubic interpolation and "LE" means LE preimage. The larger PSNR or SSIM value between the two methods is denoted in **bold**.*

(a) *T1w Reference Image*

		$r = 2$	$r = 3$	$r = 8$
BI	PSNR	27.21	24.08	18.78
	SSIM	0.925	0.853	0.639
LE	PSNR	**29.26**	**26.32**	**22.23**
	SSIM	**0.938**	**0.898**	**0.807**

(c) $r = 2$ *Interpolation*

(d) $r = 3$ *Interpolation*

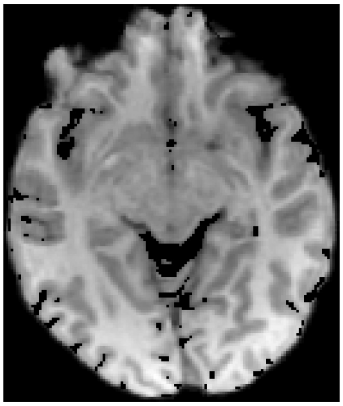
(e) $r = 8$ *Interpolation*

(f) $r = 2$ *LE MMSR*

(g) $r = 3$ *LE MMSR*

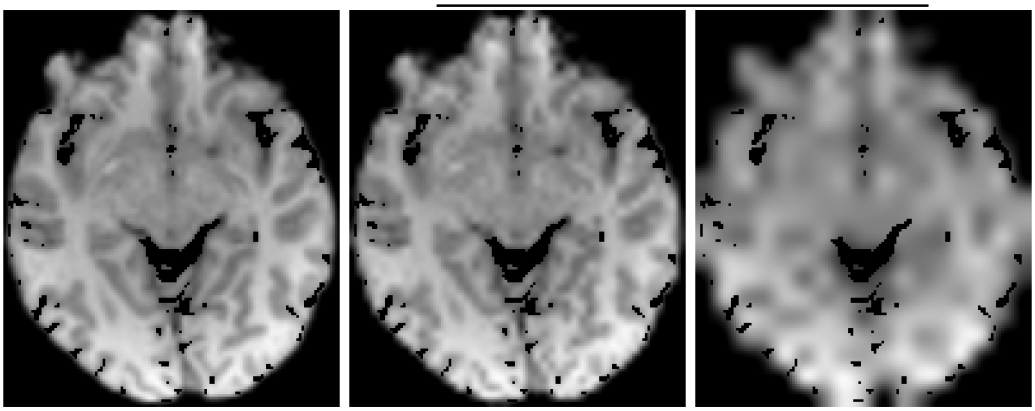
(h) $r = 8$ *LE MMSR*

Figure 7.20: *Comparison of our MMSR method and bicubic interpolation on an axial slice, BraTS volume 27, slice 63. The high resolution T1w bicubic interpolations with scaling factors $r = 2$, $r = 3$, and $r = 8$ are shown in (c), (d), and (e), respectively. The high resolution T1w LE preimages through our method using low resolution T1w images with scaling factors of $r = 2$, $r = 3$, and $r = 8$ are shown in (f), (g), and (h), respectively. In the table (b) below, "BI" means bicubic interpolation and "LE" means LE preimage. The larger PSNR or SSIM value between the two methods is denoted in* ***bold****.*

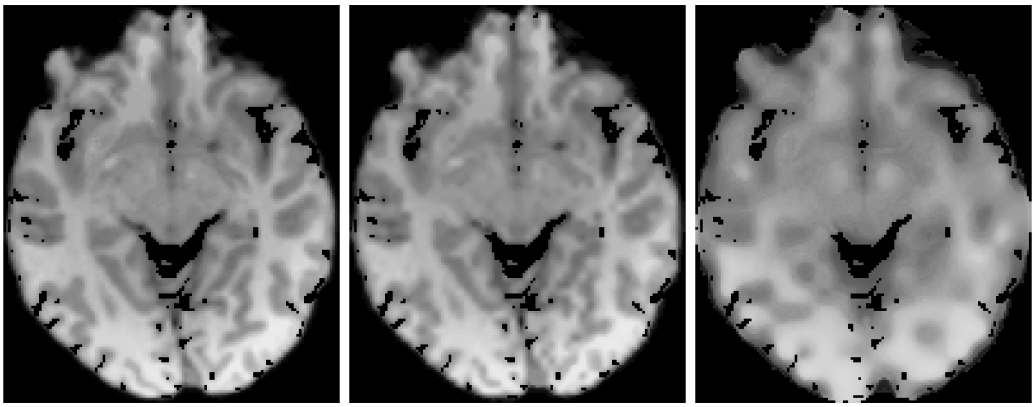


(a) *T1w Reference Image*

		$r = 2$	$r = 3$	$r = 8$
BI	PSNR	**34.56**	29.71	20.15
	SSIM	**0.976**	0.928	0.621
LE	PSNR	34.24	**30.22**	**21.52**
	SSIM	0.970	**0.937**	**0.727**



(c) $r = 2$ *Interpolation* **(d)** $r = 3$ *Interpolation* **(e)** $r = 8$ *Interpolation*



(f) $r = 2$ *LE MMSR* **(g)** $r = 3$ *LE MMSR* **(h)** $r = 8$ *LE MMSR*

7.6 Conclusion and Discussion

In conclusion, we demonstrate the ability to recover high resolution T1w images from low resolution T1w images and the use of high resolution T2w images. We show that a joint image and incorporation of spatial coordinates in LE construction result in a dramatic increase of recovered image quality versus excluding that information. Our overall results, indicated in Table 7.4 with examples given in Figures 7.17 to 7.20, also show that our method for MMSR is better-suited for situations in which the scaling factor from the low resolution image to the high resolution image is large, when compared to standard methods like bicubic interpolation. We also conduct an in-depth study of the effect of the removal of CSF from MRI brain images in processing, coming to the unexpected result regarding how that removal should be done, if at all.

Our method takes inspiration from the rotation method of Coifman and Hirn [32] for change detection. The results in Figure 7.8 also suggest that our graph matching approach is an improvement on their method for change detection, although further study would be needed to validate that. Our work also compares favorably to the similarly motivated, state-of-the-art MMSR method of Lu [83].

Table 7.5: PSNR results reported by Xiaoqiang Lu et al. in "MR image super-resolution via manifold regularized sparse learning".

Image	Non-local means	Proposed algorithm
Normal brain	32.16	**32.91**
MS brain	32.11	**32.50**

Table 7.5 shows their average PSNR for a BrainWeb volume, with a scaling factor of $r = 2$. The equivalent computation in our test is in the Normal BrainWeb section of Table 7.4. We

attribute the fact that our average PSNR for 11 slices of the same volume is higher to more detailed information informing our superresolution. The method of Lu uses small patches of pixels as data points, whereas our graph-based MMSR method assigns a data vector to each pixel.

This points to one of the major drawbacks of our method: the computational cost. Using an Intel i5-8265U processor at 1.6Ghz with 8GB of memory in MATLAB 9.13.0, the full process of creating one approximate high resolution T1w slice is about 5 minutes. The majority of that time is spent on the LE preimage calculation, specifically in the sparse optimization step used in approximating a kernel vector for a high resolution point. There may be alternate strategies for this computation which could reduce the computation time. For instance, perhaps knowledge of the neighborhood structure of the *low* resolution points can help to obtain a courser but faster approximate kernel vectors. But this would be undoubtedly a delicate balance, since accuracy of approximate kernel affects not only the learned neighborhood structure but also the weights in the neighborhood.

Finally, we consider some of the future extensions of our work. Our graph based MMSR method is certainly applicable to other MRI modalities beside T1w and T2w images, or even for different imaging methods entirely. More broadly speaking, the method is readily adaptable to other data representations such as diffusion maps. There is also potential for merging our method with neural networks. Angles and Mallat [2] use a neural network framework to recover and interpolate images from coefficients of a *scattering transform* [21], a data representation system which uses wavelets to form a kind of neural network with fixed weights. A similar scheme could be devised in the context of our graph based MMSR method, where a neural network could be trained to recover high resolution images from LE embedding coordinates.

Bibliography

[1] O. D. Abaan, E. C. Polley, S. R. Davis, Y. J. Zhu, S. Bilke, R. L. Walker, M. Pineda, Y. Gindin, Y. Jiang, W. C. Reinhold, and S. L. Holbeck. The exomes of the NCI-60 panel: A genomic resource for cancer biology and systems pharmacology. *Cancer Research*, 73, 2013.

[2] T. Angles and S. Mallat. Generative networks as inverse problems with scattering transforms. In *International Conference on Learning Representations*, 2018.

[3] D. Arthur and S. Vassilvitskii. K-means++ the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[4] W. Baatz, M. Fornasier, P. A. Markowich, and C.-B. Schönlieb. Inpainting of ancient austrian frescoes. In *Bridges Leeuwarden: Mathematics, music, art, architecture, culture*, pages 163–170, 2008.

[5] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, and C. Davatzikos. Segmentation labels and radiomic features for the pre-operative scans of the TCGA-GBM collection. *The Cancer Imaging Archive*, 2017.

[6] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, and C. Davatzikos. Segmentation labels and radiomic features for the pre-operative scans of the TCGA-LGG collection. *The Cancer Imaging Archive*, 2017.

[7] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. S. Kirby, J. B. Freymann, K. Farahani, and C. Davatzikos. Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. *Scientific Data*, 4:170117, 2017.

[8] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, R. T. Shinohara, C. Berger, S. M. Ha, M. Rozycki, et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge. *arXiv preprint arXiv:1811.02629*, 2018.

[9] S. Bauer, R. Wiest, L.-P. Nolte, and M. Reyes. A survey of MRI-based medical image analysis for brain tumor studies. *Physics in Medicine & Biology*, 58(13):R97, 2013.

[10] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[11] M. Belkin and P. Niyogi. Convergence of Laplacian eigenmaps. *Advances in Neural Information Processing Systems*, 19, 2006.

[12] M. Belkin and P. Niyogi. Convergence of Laplacian eigenmaps. In *Advances in Neural Information Processing Systems*, pages 129–136, 2007.

[13] J. Benedetto, W. Czaja, J. Dobrosotskaya, T. Doster, K. Duke, and D. Gillis. Integration of heterogeneous data for classification in hyperspectral satellite imagery. In *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVIII*, volume 8390, pages 705–716. SPIE, 2012.

[14] J. Benedetto, A. Cloninger, W. Czaja, T. Doster, K. Kochersberger, B. Manning, T. McCullough, and M. McLane. Operator based integration of information in multimodal radiological search mission with applications to anomaly detection. In *Chemical, Biological, Radiological, Nuclear, and Explosives (CBRNE) Sensing XV*, volume 9073, pages 282–290. SPIE, 2014.

[15] J. J. Benedetto and W. Czaja. Dimension reduction and remote sensing using modern harmonic analysis. *Handbook of Geomathematics*, pages 1–22, 2013.

[16] J. J. Benedetto, W. Czaja, J. Dobrosotskaya, T. Doster, and K. Duke. Spatial-spectral operator theoretic methods for hyperspectral image classification. *GEM-International Journal on Geomathematics*, 7:275–297, 2016.

[17] Y. Bengio, J.-f. Paiement, P. Vincent, O. Delalleau, N. Roux, and M. Ouimet. Out-of-sample extensions for LLE, ISOMAP, MDS, Eigenmaps, and spectral clustering. *Advances in Neural Information Processing Systems*, 16, 2003.

[18] Y. Bengio, O. Delalleau, N. L. Roux, J.-F. Paiement, P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16(10): 2197–2219, 2004.

[19] D. Besghini, M. Mauri, and R. Simonutti. Time domain NMR in polymer science: From the laboratory to the industry. *Applied Sciences*, 9(9):1801, 2019.

[20] R. W. Brown, Y.-C. N. Cheng, E. M. Haacke, M. R. Thompson, and R. Venkatesan. *Magnetic Resonance Imaging: Physical Principles and Sequence Design*. John Wiley & Sons, 2014.

[21] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, 2013.

[22] N. D. Cahill, W. Czaja, and D. W. Messinger. Schroedinger eigenmaps with nondiagonal potentials for spatial-spectral clustering of hyperspectral imagery. In *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral imagery XX*, volume 9088, pages 27–39. SPIE, 2014.

[23] C. Chen, C. Raymond, W. Speier, X. Jin, T. F. Cloughesy, D. Enzmann, B. M. Ellingson, and C. W. Arnold. Synthesizing MR image contrast enhancement using 3D high-resolution convnets. *IEEE Transactions on Biomedical Engineering*, 70(2):401–412, 2022.

[24] G. Christie, L. J. Stiltner, K. Kochersberger, M. McLean, and W. Czaja. Synchronous radiation sensing and 3D urban mapping for improved source identification. In *Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2014*, volume 9121, pages 84–91. SPIE, 2014.

[25] F. R. Chung and F. C. Graham. *Spectral Graph Theory*. Number 92. American Mathematical Soc., 1997.

[26] A. Cloninger. *Exploiting Data-dependent Structure for Improving Sensor Acquisition and Integration*. PhD thesis, University of Maryland, College Park, 2014.

[27] A. Cloninger and W. Czaja. Eigenvector localization on data-dependent graphs. In *2015 International Conference on Sampling Theory and Applications (SampTA)*, pages 608–612, 2015.

[28] A. Cloninger, W. Czaja, and T. Doster. A case study on data fusion with overlapping segments. In *2013 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–11. IEEE, 2013.

[29] A. Cloninger, W. Czaja, and T. Doster. Operator analysis and diffusion based embeddings for heterogeneous data fusion. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 1249–1252. IEEE, 2014.

[30] A. Cloninger, W. Czaja, and T. Doster. Multimodal data fusion via graph-and operator-based techniques. In *Hyperspectral Imaging and Sounding of the Environment*, pages HW2B–3. Optica Publishing Group, 2015.

[31] A. Cloninger, W. Czaja, and T. Doster. The pre-image problem for Laplacian eigenmaps utilizing L1 regularization with applications to data fusion. *Inverse Problems*, 33 (7):074006, 2017.

[32] R. R. Coifman and M. J. Hirn. Diffusion maps for changing data. *Applied and Computational Harmonic Analysis*, 36(1):79–107, 2014.

[33] D. L. Collins, A. P. Zijdenbos, V. Kollokian, J. G. Sled, N. J. Kabani, C. J. Holmes, and A. C. Evans. Design and construction of a realistic digital brain phantom. *IEEE Transactions on Medical Imaging*, 17(3):463–468, 1998.

[34] W. Czaja and M. Ehler. Schroedinger eigenmaps for the analysis of biomedical data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(5):1274–1280, 2012.

[35] W. Czaja and J. Emidih. Heterogeneous cancer cell line data fusion for identifying novel response determinants in precision medicine. In *Bioinformatics Research and Applications: 13th International Symposium, ISBRA 2017, Honolulu, HI, USA, May 29–June 2, 2017, Proceedings 13*, pages 414–419. Springer, 2017.

[36] W. Czaja, L. Moigne-Stewart, et al. Recent advances in registration, integration and fusion of remotely sensed data: Redundant representations and frames. In *Canadian Symposium on Remote Sensing*, number GSFC-E-DAA-TN15499, 2014.

[37] W. Czaja, A. Hafftka, B. Manning, and D. Weinberg. Randomized approximations of operators and their spectral decomposition for diffusion based embeddings of heterogeneous data. In *2015 3rd International Workshop on Compressed Sensing Theory and its Applications to Radar, Sonar and Remote Sensing (CoSeRa)*, pages 75–79. IEEE, 2015.

[38] W. Czaja, J. M. Murphy, and D. Weinberg. Superresolution of remotely sensed images with anisotropic features. In *2015 International Conference on Sampling Theory and Applications (SampTA)*, pages 317–321. IEEE, 2015.

[39] W. Czaja, B. Manning, L. McLean, and J. M. Murphy. Fusion of aerial gamma-ray survey and remote sensing data for a deeper understanding of radionuclide fate after radiological incidents: examples from the fukushima dai-ichi response. *Journal of Radioanalytical and Nuclear Chemistry*, 307, 2016.

[40] W. Czaja, T. Doster, and A. Halevy. An overview of numerical acceleration techniques for nonlinear dimension reduction. *Recent Applications of Harmonic Analysis to Function Spaces, Differential Equations, and Data Science: Novel Methods in Harmonic Analysis, Volume 2*, pages 797–829, 2017.

[41] W. Czaja, I. Kavalerov, and W. Li. Scattering transforms and classification of hyperspectral images. In *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XXIV*, volume 10644, pages 125–136. SPIE, 2018.

[42] W. Czaja, J. M. Murphy, and D. Weinberg. Superresolution of noisy remotely sensed images through directional representations. *IEEE Geoscience and Remote Sensing Letters*, 15(12):1837–1841, 2018.

[43] W. Czaja, D. Dong, P.-E. Jabin, and F. O. Ndjakou Njeunje. Transport model for feature extraction. *SIAM Journal on Mathematics of Data Science*, 3(1):321–341, 2021.

[44] X. Dai, Y. Lei, Y. Fu, W. J. Curran, T. Liu, H. Mao, and X. Yang. Multimodal MRI synthesis using unified generative adversarial networks. *Medical Physics*, 47(12):6343–6354, 2020.

[45] W. Dong, L. Zhang, G. Shi, and X. Wu. Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. *IEEE Transactions on Image Processing*, 20(7):1838–1857, 2011.

[46] T. J. Doster. *Harmonic Analysis Inspired Data Fusion for Applications in Remote Sensing*. PhD thesis, University of Maryland, College Park, 2014.

[47] O. J. Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

[48] M. Ehler, V. Rajapakse, B. Zeeberg, B. Brooks, J. Brown, W. Czaja, and R. F. Bonner. Analysis of temporal-spatial co-variation within gene expression microarray data in an organogenesis model. In *Bioinformatics Research and Applications: 6th International Symposium, ISBRA 2010, Storrs, CT, USA, May 23-26, 2010. Proceedings 6*, pages 38–49. Springer, 2010.

[49] Z. A. S. Emam. *Towards an Efficient Semantic Segmentation Pipeline for 3D Electron Microscopy Data*. PhD thesis, University of Maryland, College Park, 2022.

[50] R. Everson. Orthogonal, but not orthonormal, procrustes problems. *Advances in Computational Mathematics*, 3(4), 1998.

[51] D. Eynard, A. Kovnatsky, and M. M. Bronstein. Laplacian colormaps: a framework for structure-preserving color transformations. In *Computer Graphics Forum*, volume 33, pages 215–224. Wiley Online Library, 2014.

[52] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.-C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, et al. 3D Slicer as an image computing platform for the quantitative imaging network. *Magnetic Resonance Imaging*, 30(9): 1323–1341, 2012.

[53] J. C. Flake. *The Multiplicative Zak transform, Dimension Reduction, and Wavelet Analysis of LIDAR data*. PhD thesis, University of Maryland, College Park, 2010.

[54] E. W. Forgy. Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[55] L. Fowl. *Analysis of Data Security Vulnerabilities in Deep Learning*. PhD thesis, University of Maryland, College Park, 2022.

[56] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, Sep. 1977. ISSN 0098-3500.

[57] M. J. Garnett, E. J. Edelman, S. J. Heidorn, C. D. Greenman, A. Dastur, K. W. Lau, P. Greninger, I. R. Thompson, X. Luo, J. Soares, and Q. Liu. Systematic identification of genomic markers of drug sensitivity in cancer cells. *Nature*, 483, 2012.

[58] K. Glashoff and C. P. Ortlieb. Composition operators, matrix representation, and the finite section method: A theoretical framework for maps between shapes. *arXiv preprint arXiv:1705.00325*, 2017.

[59] M. Goldblum. *Adversarial Robustness and Robust Meta-Learning for Neural Networks*. PhD thesis, University of Maryland, College Park, 2020.

[60] M. Goldblum, L. Fowl, and W. Czaja. Sheared multi-scale weight sharing for multispectral superresolution. In *Algorithms, Technologies, and Applications for Multispectral and Hyperspectral Imagery XXV*, volume 10986, pages 263–271. SPIE, 2019.

[61] J. C. Gower. Generalized procrustes analysis. *Psychometrika*, 40:33–51, 1975.

[62] A. Halevy. *Extensions of Laplacian eigenmaps for manifold learning*. PhD thesis, University of Maryland, College Park, 2011.

[63] R. H. Hashemi, W. G. Bradley, Jr, and C. J. Lisanti. *MRI : The Basics*. Wolters Kluwer Health, Philadelphia, fourth edition, 2018.

[64] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, 2009.

[65] A. Hore and D. Ziou. Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369. IEEE, 2010.

[66] W. Huang, T. A. Bolton, J. D. Medaglia, D. S. Bassett, A. Ribeiro, and D. Van De Ville. Graph signal processing of human brain imaging data. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 980–984. IEEE, 2018.

[67] K. Karhunen. Zur spektraltheorie stochastischer prozesse. *Ann. Acad. Sci. Fennicae, AI*, 34, 1946.

[68] I. Kavalerov. *Impact Of Semantics, Physics And Adversarial Mechanisms In Deep Learning*. PhD thesis, University of Maryland, College Park, 2020.

[69] A. Kovnatsky, M. M. Bronstein, X. Bresson, and P. Vandergheynst. Functional correspondence by matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 905–914, 2015.

[70] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[71] R. K. S. Kwan, A. C. Evans, and G. B. Pike. An extensible MRI simulator for post-processing evaluation. In *Visualization in Biomedical Computing: 4th International Conference, VBC'96 Hamburg, Germamy, September 22–25, 1996 Proceedings*, pages 135–140. Springer, 1996.

[72] R.-S. Kwan, A. C. Evans, and G. B. Pike. MRI simulation-based evaluation of image-processing and classification methods. *IEEE Transactions on Medical Imaging*, 18(11): 1085–1097, 1999.

[73] J.-Y. Kwok and I.-H. Tsang. The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, 15(6):1517–1525, 2004.

[74] J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

[75] M. Leordeanu, M. Hebert, and R. Sukthankar. An integer projected fixed point method for graph matching and map inference. In *Advances in Neural Information Processing Systems*, pages 1114–1122, 2009.

[76] R. Levie, E. Isufi, and G. Kutyniok. On the transferability of spectral graph filters. In *2019 13th International Conference on Sampling Theory and Applications (SampTA)*, pages 1–5. IEEE, 2019.

[77] H. Li, B. Manjunath, and S. K. Mitra. Multisensor image fusion using the wavelet transform. *Graphical Models and Image Processing*, 57(3):235–245, 1995.

[78] W. Li. *Topics in Harmonic Analysis, Sparse Representations, and Data Analysis*. PhD thesis, University of Maryland, College Park, 2018.

[79] Y. Li. *Feature extraction in image processing and deep learning*. PhD thesis, University of Maryland, College Park, 2018.

[80] J. Limb, C. Rubinstein, and J. Thompson. Digital coding of color video signals-a review. *IEEE Transactions on Communications*, 25(11):1349–1385, 1977.

[81] O. Lindenbaum, M. Salhov, A. Yeredor, and A. Averbuch. Gaussian bandwidth selection for manifold learning and classification. *Data Mining and Knowledge Discovery*, 34: 1676–1712, 2020.

[82] M. Loeve. Fonctions aléatoires du second ordre, supplement to p. *Lévy, Processus Stochastiques et Mouvement Brownien, Gauthier-Villars, Paris*, 1948.

[83] X. Lu, Z. Huang, and Y. Yuan. MR image super-resolution via manifold regularized sparse learning. *Neurocomputing*, 162:96–104, 2015. ISSN 0925-2312.

[84] A. Luna, V. N. Rajapakse, F. G. Sousa, J. Gao, N. Schultz, S. Varma, W. Reinhold, C. Sander, and Y. Pommier. rcellminer: exploring molecular profiles and drug response of the NCI-60 cell lines in R. *Bioinformatics*, 32(8):1272–1274, 2016.

[85] Q. Lyu, H. Shan, C. Steber, C. Helis, C. Whitlow, M. Chan, and G. Wang. Multi-contrast super-resolution MRI through a progressive network. *IEEE Transactions on Medical Imaging*, 39(9):2738–2749, 2020.

[86] J. V. Manjón, P. Coupé, A. Buades, D. L. Collins, and M. Robles. MRI superresolution using self-similarity and image priors. *Journal of Biomedical Imaging*, 2010:1–11, 2010.

[87] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, et al. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, 2014.

[88] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, et al. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, 2014.

[89] H. Mhaskar. A unified framework for harmonic analysis of functions on directed graphs and changing data. *Applied and Computational Harmonic Analysis*, 44(3):611–644, 2018. ISSN 1063-5203.

[90] J. M. Murphy. *Anisotropic harmonic analysis and integration of remotely sensed data*. PhD thesis, University of Maryland, College Park, 2015.

[91] F. O. N. Njeunje. *Computational methods in machine learning: transport model, Haar wavelet, DNA classification, and MRI*. PhD thesis, University of Maryland, College Park, 2018.

[92] S. Olsen, R. Gold, A. Gooch, and B. Gooch. Recovering color from black and white photographs. In *2010 IEEE International Conference on Computational Photography (ICCP)*, pages 1–8. IEEE, 2010.

[93] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (ToG)*, 31(4):1–11, 2012.

[94] K. Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

[95] N. Perraudin, B. Ricaud, D. I. Shuman, and P. Vandergheynst. Global and local uncertainty principles for signals on graphs. *APSIPA Transactions on Signal and Information Processing*, 7, 2018.

[96] E. Plenge, D. H. Poot, M. Bernsen, G. Kotek, G. Houston, P. Wielopolski, L. van der Weerd, W. J. Niessen, and E. Meijering. Super-resolution methods in MRI: can they improve the trade-off between resolution, signal-to-noise ratio, and acquisition time? *Magnetic Resonance in Medicine*, 68(6):1983–1993, 2012.

[97] Y. Pommier. Drugging topoisomerases: lessons and challenges. *ACS Chemical Biology*, 8, 2013.

[98] G. Puy, N. Tremblay, R. Gribonval, and P. Vandergheynst. Random sampling of bandlimited signals on graphs. *Applied and Computational Harmonic Analysis*, 44(2):446–475, 2018.

[99] V. N. Rajapakse, W. Czaja, Y. G. Pommier, W. C. Reinhold, and S. Varma. Predicting expression-related features of chromosomal domain organization with network-structured analysis of gene expression and chromosomal location. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pages 226–233, 2012.

[100] M. G. Rees, B. Seashore-Ludlow, J. H. Cheah, D. J. Adams, E. V. Price, S. Gill, S. Javaid, M. E. Coletti, V. L. Jones, N. E. Bodycombe, and C. K. Soule. Correlating chemical sensitivity and basal gene expression reveals mechanism of action. *Nature Chemical Biology*, 12, 2016.

[101] W. C. Reinhold, M. Sunshine, S. Varma, J. H. Doroshow, and Y. Pommier. Using cellminer 1.6 for systems pharmacology and genomic analysis of the NCI-60. *Clinical Cancer Research*, 21(17):3841–3852, 2015.

[102] N. Roussopoulos, S. Kelley, and F. Vincent. Nearest neighbor queries. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 71–79, 1995.

[103] D. L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5(4):517, 1994.

[104] A. Sakiyama, Y. Tanaka, T. Tanaka, and A. Ortega. Eigendecomposition-free sampling set selection for graph signals. *IEEE Transactions on Signal Processing*, 67(10):2679–2692, 2019.

[105] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[106] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In *Artificial Neural Networks—ICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceeedings*, pages 583–588. Springer, 2005.

[107] P. H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, Mar 1966. ISSN 1860-0980.

[108] A. Sharma and G. Hamarneh. Missing MRI pulse sequence synthesis using multi-modal generative adversarial network. *IEEE Transactions on Medical Imaging*, 39(4):1170–1183, 2019.

[109] J. Shi and J. Malik. Normalized cuts and image segmentation. *Departmental Papers (CIS)*, page 107, 2000.

[110] D. I. Shuman, B. Ricaud, and P. Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260–291, 2016.

[111] F. G. Sousa, R. Matuo, S. W. Tang, V. N. Rajapakse, A. Luna, C. Sander, S. Varma, P. H. Simon, J. H. Doroshow, W. C. Reinhold, and Y. Pommier. Alterations of DNA repair genes in the NCI-60 cell lines and their predictive value for anticancer drug activity. *DNA Repair*, 28, 2015.

[112] D. A. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.

[113] W. Sun, A. Halevy, J. J. Benedetto, W. Czaja, W. Li, C. Liu, B. Shi, and R. Wang. Nonlinear dimensionality reduction via the ENH-LTSA method for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(2):375–388, 2013.

[114] W. Sun, A. Halevy, J. J. Benedetto, W. Czaja, C. Liu, H. Wu, B. Shi, and W. Li. UL-Isomap based nonlinear dimensionality reduction for hyperspectral imagery classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 89:25–36, 2014.

[115] S. W. Tang, S. Bilke, L. Cao, J. Murai, F. G. Sousa, M. Yamade, V. Rajapakse, S. Varma, L. J. Helman, J. Khan, and P. S. Meltzer. SLFN11 is a transcriptional target of EWS-FLI1 and a determinant of drug response in Ewing Sarcoma. *Clinical Cancer Research*, 21, 2015.

[116] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4): 401–419, 1952.

[117] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[118] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst. Compressive spectral clustering. In *International Conference on Machine Learning*, pages 1002–1011, 2016.

[119] J. T. Vogelstein, J. M. Conroy, V. Lyzinski, L. J. Podrazik, S. G. Kratzer, E. T. Harley, D. E. Fishkind, R. J. Vogelstein, and C. E. Priebe. Fast approximate quadratic programming for graph matching. *PLOS ONE*, 10(4):e0121002, 2015.

[120] D. Weinberg. *Multiscale and directional representations of high-dimensional information content in remotely sensed data*. PhD thesis, University of Maryland, College Park, 2015.

[121] D. P. Widemann. *Dimensionality reduction for hyperspectral data*. University of Maryland, College Park, 2008.

[122] C. Williams. On a connection between kernel PCA and metric multidimensional scaling. *Advances in Neural Information Processing Systems*, 13, 2000.

[123] C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems*, 13, 2000.

[124] H. Zheng, K. Zeng, D. Guo, J. Ying, Y. Yang, X. Peng, F. Huang, Z. Chen, and X. Qu. Multi-contrast brain MRI image super-resolution with gradient-guided edge enhancement. *IEEE Access*, 6:57856–57867, 2018.

[125] B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, and M. S. Rosen. Image reconstruction by domain-transform manifold learning. *Nature*, 555(7697):487–492, 2018.

[126] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 912–919, 2003.

[127] G. Zoppoli, M. Regairaz, E. Leo, W. C. Reinhold, S. Varma, A. Ballestrero, J. H. Doroshow, and Y. Pommier. PutativeDNA/RNA helicase Schlafen-11 (SLFN11) sensitizes cancer cells to DNA-damaging agents. *Proceedings of the National Academy of Sciences*, 109, 2012.