Chapter 6

Large-scale Manifold Learning

Ameet Talwalkar,[1] Sanjiv Kumar,[2] Mehryar Mohri,[3] Henry Rowley[4]

6.1 Introduction

The problem of dimensionality reduction arises in many computer vision applications, where it is natural to represent images as vectors in a high-dimensional space. Manifold learning techniques extract low-dimensional structure from high-dimensional data in an unsupervised manner. These techniques typically try to unfold the underlying manifold so that some quantity, e.g., pairwise geodesic distances, is maintained invariant in the new space. This makes certain applications such as K-means clustering more effective in the transformed space.

In contrast to linear dimensionality reduction techniques such as Principal Component Analysis (PCA), manifold learning methods provide more powerful non-linear dimensionality reduction by preserving the local structure of the input data. Instead of assuming global linearity, these methods typically make a weaker local-linearity assumption, i.e., for nearby points in high-dimensional input space, l_2 distance is assumed to be a good measure of geodesic distance, or distance along the manifold. Good sampling of the underlying manifold is essential for this assumption to hold. In fact, many manifold learning techniques provide guarantees that the accuracy of the recovered manifold increases as the number of data samples increases. In the limit of infinite samples, one can recover the true underlying manifold for certain classes of manifolds (Tenenbaum et al., 2000; Belkin & Niyogi, 2006; Donoho & Grimes, 2003). However, there is

[1]University of California Berkeley, Berkeley, CA, *ameet@eecs.berkeley.edu*.
[2]Google Research, New York, NY, *sanjivk@google.com*.
[3]Courant Institute and Google Research, New York, NY, *mohri@cs.nyu.edu*.
[4]Google Research, Mountain View, CA, *har@google.com*.

a trade-off between improved sampling of the manifold and the computational cost of manifold learning algorithms. In this chapter, we address the computational challenges involved in learning manifolds given millions of face images extracted from the Web.

Several manifold learning techniques have been proposed, e.g., Semidefinite Embedding (SDE) (Weinberger & Saul, 2006), Isomap (Tenenbaum et al., 2000), Laplacian Eigenmaps (Belkin & Niyogi, 2001) and Local Linear Embedding (LLE) (Roweis & Saul, 2000). SDE aims to preserve distances and angles between all neighboring points. It is formulated as an instance of semidefinite programming, and is thus prohibitively expensive for large-scale problems. Isomap constructs a dense matrix of approximate geodesic distances between *all* pairs of inputs, and aims to find a low dimensional space that best preserves these distances. Other algorithms, e.g., Laplacian Eigenmaps and LLE, focus only on preserving local neighborhood relationships in the input space. They generate low-dimensional representations via manipulation of the graph Laplacian or other sparse matrices related to the graph Laplacian (Chapelle et al., 2006). In this chapter, we focus mainly on Isomap and Laplacian Eigenmaps, as both methods have good theoretical properties and the differences in their approaches allow us to make interesting comparisons between dense and sparse methods.

All of the manifold learning methods described above can be viewed as specific instances of Kernel PCA (Ham et al., 2004). These kernel-based algorithms require SVD of matrices of size $n \times n$, where n is the number of samples. This generally takes $O(n^3)$ time. When only a few singular values and singular vectors are required, there exist less computationally intensive techniques such as Jacobi, Arnoldi, Hebbian and more recent randomized methods (Golub & Loan, 1983; Gorrell, 2006; Rokhlin et al., 2009). These iterative methods require computation of matrix-vector products at each step and involve multiple passes through the data. When the matrix is sparse, these techniques can be implemented relatively efficiently. However, when dealing with a large, dense matrix, as in the case of Isomap, these products become expensive to compute. Moreover, when working with 18M data points, it is not possible even to store the full 18M $\times$ 18M matrix ($\sim$1300TB), rendering the iterative methods infeasible. Random sampling techniques provide a powerful alternative for approximate SVD and only operate on a subset of the matrix.

In this chapter, we examine both the Nyström and Column sampling methods (defined in Section 6.3), providing the first direct comparison between their performances on practical applications. The Nyström approximation has been studied in the machine learning community (Williams & Seeger, 2000) (Drineas & Mahoney, 2005). In parallel, Column sampling techniques have been analyzed in the theoretical Computer Science community (Frieze et al., 1998; Drineas et al., 2006; Deshpande et al., 2006). However, prior to initial work in Talwalkar et al. (2008); Kumar et al. (2009a), the relationship between these approximations had not been well studied. We provide an extensive analysis of these algorithms, show connections between these approximations and provide a direct comparison between their performances.

Apart from singular value decomposition, the other main computational hurdle associated with Isomap and Laplacian Eigenmaps is large-scale graph construction and manipulation. These algorithms first need to construct a local neighborhood graph in the input space, which is an $O(n^2)$ problem given n data points. Moreover, Isomap requires shortest paths between every pair of points requiring $O(n^2 \log n)$ computation. Both of these steps become intractable when n is as large as 18M. In this study, we use approximate nearest neighbor methods, and explore random sampling based SVD that requires the computation of shortest paths only for a subset of points. Furthermore, these approximations allow for an efficient distributed implementation of the algorithms.

We now summarize our main contributions. First, we present the largest scale study so far on manifold learning, using 18M data points. To date, the largest manifold learning study involves the analysis of music data using 267K points (Platt, 2004). In vision, the largest study is limited to less than 10K images (He et al., 2005). Second, we show connections between two random sampling based singular value decomposition algorithms and provide the first direct comparison of their performances. Finally, we provide a quantitative comparison of Isomap and Laplacian Eigenmaps for large-scale face manifold construction on clustering and classification tasks.

6.2 Background

In this section, we introduce notation (summarized in Table 6.1) and present basic definitions of two of the most common sampling-based techniques for matrix approximation.

Table 6.1: Summary of notation used throughout this chapter. See Section 6.2.1 for further details.

$\mathbf{T}$	arbitrary matrix in $\mathbb{R}^{a \times b}$
$\mathbf{T}^{(j)}$	jth column vector of $\mathbf{T}$ for $j = 1 \ldots b$
$\mathbf{T}_{(i)}$	ith row vector of $\mathbf{T}$ for $i = 1 \ldots a$
$\mathbf{T}^{(i:j)}, \mathbf{T}_{(i:j)}$	ith through jth columns / rows of $\mathbf{T}$
$\mathbf{T}_k$	'best' rank-k approximation to $\mathbf{T}$
$\\|\cdot\\|_2, \\|\cdot\\|_F$	Spectral, Frobenius norms of $\mathbf{T}$
$\mathbf{v}$	arbitrary vector in $\mathbb{R}^a$
$\\|\cdot\\|$	l_2 norm of a vector
$\mathbf{T} = \mathbf{U}_T \mathbf{\Sigma}_T \mathbf{V}_T^\top$	Singular Value Decomposition (SVD) of $\mathbf{T}$
$\mathbf{K}$	SPSD kernel matrix in $\mathbb{R}^{n \times n}$ with rank$(\mathbf{K}) = r \leq n$
$\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$	SVD of $\mathbf{K}$
$\mathbf{K}^+$	Pseudo-inverse of $\mathbf{K}$
$\widetilde{\mathbf{K}}$	approximation to $\mathbf{K}$ derived from $l \ll n$ of its columns

6.2.1 Notation

Let $\mathbf{T} \in \mathbb{R}^{a \times b}$ be an arbitrary matrix. We define $\mathbf{T}^{(j)}$, $j = 1 \dots b$, as the jth column vector of $\mathbf{T}$, $\mathbf{T}_{(i)}$, $i = 1 \dots a$, as the ith row vector of $\mathbf{T}$ and $\|\cdot\|$ the l_2 norm of a vector. Furthermore, $\mathbf{T}^{(i:j)}$ refers to the ith through jth columns of $\mathbf{T}$ and $\mathbf{T}_{(i:j)}$ refers to the ith through jth rows of $\mathbf{T}$. We denote by $\mathbf{T}_k$ the 'best' rank-k approximation to $\mathbf{T}$, i.e., $\mathbf{T}_k = \operatorname{argmin}_{\mathbf{V} \in \mathbb{R}^{a \times b}, \operatorname{rank}(\mathbf{V})=k} \|\mathbf{T} - \mathbf{V}\|_\xi$, where $\xi \in \{2, F\}$ and $\|\cdot\|_2$ denotes the spectral norm and $\|\cdot\|_F$ the Frobenius norm of a matrix. Assuming that $\operatorname{rank}(\mathbf{T}) = r$, we can write the compact Singular Value Decomposition (SVD) of this matrix as $\mathbf{T} = \mathbf{U}_T \mathbf{\Sigma}_T \mathbf{V}_T^\top$ where $\mathbf{\Sigma}_T$ is diagonal and contains the singular values of $\mathbf{T}$ sorted in decreasing order and $\mathbf{U}_T \in \mathbb{R}^{a \times r}$ and $\mathbf{V}_T \in \mathbb{R}^{b \times r}$ have orthogonal columns that contain the left and right singular vectors of $\mathbf{T}$ corresponding to its singular values. We can then describe $\mathbf{T}_k$ in terms of its SVD as $\mathbf{T}_k = \mathbf{U}_{T,k} \mathbf{\Sigma}_{T,k} \mathbf{V}_{T,k}^\top$ where $\mathbf{\Sigma}_{T,k}$ is a diagonal matrix of the top k singular values of $\mathbf{T}$ and $\mathbf{U}_{T,k}$ and $\mathbf{V}_{T,k}$ are the associated left and right singular vectors.

Now let $\mathbf{K} \in \mathbb{R}^{n \times n}$ be a symmetric positive semidefinite (SPSD) kernel or Gram matrix with $\operatorname{rank}(\mathbf{K}) = r \leq n$, i.e. a symmetric matrix for which there exists an $\mathbf{X} \in \mathbb{R}^{N \times n}$ such that $\mathbf{K} = \mathbf{X}^\top \mathbf{X}$. We will write the SVD of $\mathbf{K}$ as $\mathbf{K} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$, where the columns of $\mathbf{U}$ are orthogonal and $\mathbf{\Sigma} = \operatorname{diag}(\sigma_1, \dots, \sigma_r)$ is diagonal. The pseudo-inverse of $\mathbf{K}$ is defined as $\mathbf{K}^+ = \sum_{t=1}^r \sigma_t^{-1} \mathbf{U}^{(t)} {\mathbf{U}^{(t)}}^\top$, and $\mathbf{K}^+ = \mathbf{K}^{-1}$ when $\mathbf{K}$ is full rank. For $k < r$, $\mathbf{K}_k = \sum_{t=1}^k \sigma_t \mathbf{U}^{(t)} {\mathbf{U}^{(t)}}^\top = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{U}_k^\top$ is the 'best' rank-k approximation to $\mathbf{K}$, i.e., $\mathbf{K}_k = \operatorname{argmin}_{\mathbf{K}' \in \mathbb{R}^{n \times n}, \operatorname{rank}(\mathbf{K}')=k} \|\mathbf{K} - \mathbf{K}'\|_{\xi \in \{2,F\}}$, with $\|\mathbf{K} - \mathbf{K}_k\|_2 = \sigma_{k+1}$ and $\|\mathbf{K} - \mathbf{K}_k\|_F = \sqrt{\sum_{t=k+1}^r \sigma_t^2}$ (Golub & Loan, 1983).

We will be focusing on generating an approximation $\widetilde{\mathbf{K}}$ of $\mathbf{K}$ based on a sample of $l \ll n$ of its columns. We assume that we sample columns uniformly without replacement as suggested by Kumar et al. (2009b), though various methods have been proposed to select columns (see Chapter 4 of Talwalkar (2010) for more details on various sampling schemes). Let $\mathbf{C}$ denote the $n \times l$ matrix formed by these columns and $\mathbf{W}$ the $l \times l$ matrix consisting of the intersection of these l columns with the corresponding l rows of $\mathbf{K}$. Note that $\mathbf{W}$ is SPSD since $\mathbf{K}$ is SPSD. Without loss of generality, the columns and rows of $\mathbf{K}$ can be rearranged based on this sampling so that $\mathbf{K}$ and $\mathbf{C}$ be written as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{W} & \mathbf{K}_{21}^\top \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{K}_{21} \end{bmatrix}. \tag{6.1}$$

The approximation techniques discussed next use the SVD of $\mathbf{W}$ and $\mathbf{C}$ to generate approximations for $\mathbf{K}$.

6.2.2 Nyström method

The Nyström method was first introduced as a quadrature method for numerical integration, used to approximate eigenfunction solutions (Nyström, 1928; Baker, 1977). More recently, it was presented in Williams and Seeger (2000) to speed up kernel algorithms and has been used in applications ranging from manifold learning to image segmentation (Platt, 2004; Fowlkes et al., 2004; Talwalkar et al., 2008). The Nyström method uses $\mathbf{W}$ and $\mathbf{C}$ from (6.1) to approximate $\mathbf{K}$. Assuming a uniform sampling of the columns, the Nyström method generates a rank-k approximation $\widetilde{\mathbf{K}}$ of $\mathbf{K}$ for $k < n$ defined by:

$$\widetilde{\mathbf{K}}_k^{nys} = \mathbf{C}\mathbf{W}_k^{+}\mathbf{C}^\top \approx \mathbf{K}, \tag{6.2}$$

where $\mathbf{W}_k$ is the best k-rank approximation of $\mathbf{W}$ with respect to the spectral or Frobenius norm and $\mathbf{W}_k^{+}$ denotes the pseudo-inverse of $\mathbf{W}_k$. If we write the SVD of $\mathbf{W}$ as $\mathbf{W} = \mathbf{U}_W\mathbf{\Sigma}_W\mathbf{U}_W^\top$, then from (6.2) we can write

$$\begin{aligned}\widetilde{\mathbf{K}}_k^{nys} &= \mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^{+}\mathbf{U}_{W,k}^\top\mathbf{C}^\top \\ &= \left(\sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^{+}\right)\left(\frac{n}{l}\mathbf{\Sigma}_{W,k}\right)\left(\sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^{+}\right)^\top,\end{aligned}$$

and hence the Nyström method approximates the top k singular values ($\mathbf{\Sigma}_k$) and singular vectors ($\mathbf{U}_k$) of $\mathbf{K}$ as:

$$\widetilde{\mathbf{\Sigma}}_{nys} = \left(\frac{n}{l}\right)\mathbf{\Sigma}_{W,k} \quad \text{and} \quad \widetilde{\mathbf{U}}_{nys} = \sqrt{\frac{l}{n}}\mathbf{C}\mathbf{U}_{W,k}\mathbf{\Sigma}_{W,k}^{+}. \tag{6.3}$$

Since the running time complexity of compact SVD on $\mathbf{W}$ is in $O(l^2k)$ and matrix multiplication with $\mathbf{C}$ takes $O(nlk)$, the total complexity of the Nyström approximation computation is in $O(nlk)$.

6.2.3 Column sampling method

The Column sampling method was introduced to approximate the SVD of any rectangular matrix (Frieze et al., 1998). It generates approximations of $\mathbf{K}$ by using the SVD of $\mathbf{C}$.[5] If we write the SVD of $\mathbf{C}$ as $\mathbf{C} = \mathbf{U}_C\mathbf{\Sigma}_C\mathbf{V}_C^\top$ then the Column sampling method approximates the top k singular values ($\mathbf{\Sigma}_k$) and singular vectors ($\mathbf{U}_k$) of $\mathbf{K}$ as:

$$\widetilde{\mathbf{\Sigma}}_{col} = \sqrt{\frac{n}{l}}\mathbf{\Sigma}_{C,k} \quad \text{and} \quad \widetilde{\mathbf{U}}_{col} = \mathbf{U}_C = \mathbf{C}\mathbf{V}_{C,k}\mathbf{\Sigma}_{C,k}^{+}. \tag{6.4}$$

The runtime of the Column sampling method is dominated by the SVD of $\mathbf{C}$. The algorithm takes O(nlk) time to perform compact SVD on $\mathbf{C}$, but is still more expensive than the Nyström method as the constants for SVD are greater than those for the O(nlk) matrix multiplication step in the Nyström method.

[5]The Nyström method also uses sampled columns of $\mathbf{K}$, but the Column sampling method is named so because it uses direct decomposition of $\mathbf{C}$, while the Nyström method decomposes its submatrix, $\mathbf{W}$.

6.3 Comparison of sampling methods

Given that two sampling-based techniques exist to approximate the SVD of SPSD matrices, we pose a natural question: which method should one use to approximate singular values, singular vectors and low-rank approximations? We first analyze the form of these approximations and then empirically evaluate their performance in Section 6.3.3 on a variety of datasets.

6.3.1 Singular values and singular vectors

As shown in (6.3) and (6.4), the singular values of $\mathbf{K}$ are approximated as the scaled singular values of $\mathbf{W}$ and $\mathbf{C}$, respectively. The scaling terms are quite rudimentary and are primarily meant to *compensate* for the 'small sample size' effect for both approximations. Formally, these scaling terms make the approximations in (6.3) and (6.4) unbiased estimators of the true singular values. The form of singular vectors is more interesting. The Column sampling singular vectors ($\widetilde{\mathbf{U}}_{col}$) are orthonormal since they are the singular vectors of $\mathbf{C}$. In contrast, the Nyström singular vectors ($\widetilde{\mathbf{U}}_{nys}$) are approximated by *extrapolating* the singular vectors of $\mathbf{W}$ as shown in (6.3), and are *not* orthonormal. It is easy to verify that $\widetilde{\mathbf{U}}_{nys}^\top \widetilde{\mathbf{U}}_{nys} \neq \mathbf{I}_l$, where $\mathbf{I}_l$ is the identity matrix of size l. As we show in Section 6.3.3, this adversely affects the accuracy of singular vector approximation from the Nyström method.

It is possible to orthonormalize the Nyström singular vectors by using QR decomposition. Since $\widetilde{\mathbf{U}}_{nys} \propto \mathbf{C}\mathbf{U}_W \mathbf{\Sigma}_W^+$, where $\mathbf{U}_W$ is orthogonal and $\mathbf{\Sigma}_W$ is diagonal, this simply implies that QR decomposition creates an orthonormal span of $\mathbf{C}$ rotated by $\mathbf{U}_W$. However, the complexity of QR decomposition of $\widetilde{\mathbf{U}}_{nys}$ is the same as that of the SVD of $\mathbf{C}$. Thus, the computational cost of orthogonalizing $\widetilde{\mathbf{U}}_{nys}$ would nullify the computational benefit of the Nyström method over Column sampling.

6.3.2 Low-rank approximation

Several studies have empirically shown that the accuracy of low-rank approximations of kernel matrices is tied to the performance of kernel-based learning algorithms (Williams & Seeger, 2000; Talwalkar et al., 2008; Zhang et al., 2008). Furthermore, the effect of an approximation in the kernel matrix on the *hypothesis* generated by several widely used kernel-based learning algorithms has been theoretically analyzed (Cortes et al., 2010). Hence, accurate low-rank approximations are of great practical interest in machine learning. As discussed in Section 6.2.1, the optimal $\mathbf{K}_k$ is given by,

$$\mathbf{K}_k = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{U}_k^\top = \mathbf{U}_k \mathbf{U}_k^\top \mathbf{K} = \mathbf{K}\mathbf{U}_k \mathbf{U}_k^\top \tag{6.5}$$

where the columns of $\mathbf{U}_k$ are the k singular vectors of $\mathbf{K}$ corresponding to the top k singular values of $\mathbf{K}$. We refer to $\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{U}_k^\top$ as *Spectral Reconstruction*, since it uses both the singular values and vectors of $\mathbf{K}$, and $\mathbf{U}_k \mathbf{U}_k^\top \mathbf{K}$ as *Matrix*

Projection, since it uses only singular vectors to compute the projection of $\mathbf{K}$ onto the space spanned by vectors $\mathbf{U}_k$. These two low-rank approximations are equal only if $\mathbf{\Sigma}_k$ and $\mathbf{U}_k$ contain the true singular values and singular vectors of $\mathbf{K}$. Since this is not the case for approximate methods such as Nyström and Column sampling these two measures generally give different errors. From an application point of view, matrix projection approximations, although they can be quite accurate, are not necessarily symmetric and require storage of and multiplication with $\mathbf{K}$. Hence, although matrix projection is often analyzed theoretically, for large-scale problems, the storage and computational requirements may be inefficient or even infeasible. As such, in the context of large-scale manifold learning, we focus on spectral reconstructions in this chapter (for further discussion on matrix projection, see Kumar et al. (2009a)).

Using (6.3), the Nyström spectral reconstruction is:

$$\widetilde{\mathbf{K}}_k^{nys} = \widetilde{\mathbf{U}}_{nys,k}\widetilde{\mathbf{\Sigma}}_{nys,k}\widetilde{\mathbf{U}}_{nys,k}^\top = \mathbf{C}\mathbf{W}_k^+\mathbf{C}^\top. \tag{6.6}$$

When $k = l$, this approximation perfectly reconstructs three blocks of $\mathbf{K}$, and $\mathbf{K}_{22}$ is approximated by the Schur Complement of $\mathbf{W}$ in $\mathbf{K}$:

$$\widetilde{\mathbf{K}}_l^{nys} = \mathbf{C}\mathbf{W}^+\mathbf{C}^\top = \begin{bmatrix} \mathbf{W} & \mathbf{K}_{21}^\top \\ \mathbf{K}_{21} & \mathbf{K}_{21}\mathbf{W}^+\mathbf{K}_{21} \end{bmatrix}. \tag{6.7}$$

The Column sampling spectral reconstruction has a similar form as (6.6):

$$\widetilde{\mathbf{K}}_k^{col} = \widetilde{\mathbf{U}}_{col,k}\widetilde{\mathbf{\Sigma}}_{col,k}\widetilde{\mathbf{U}}_{col,k}^\top = \sqrt{n/l}\,\mathbf{C}\big((\mathbf{C}^\top\mathbf{C})_k^{\frac{1}{2}}\big)^+\mathbf{C}^\top. \tag{6.8}$$

Note that a scaling term appears in the Column sampling reconstruction. To analyze the two approximations, we consider an alternative characterization using the fact that $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$ for some $\mathbf{X} \in \mathbb{R}^{N\times n}$. Similar to Drineas and Mahoney (2005), we define a zero-one sampling matrix, $\mathbf{S} \in \mathbb{R}^{n\times l}$, that selects l columns from $\mathbf{K}$, i.e., $\mathbf{C} = \mathbf{KS}$. Each column of $\mathbf{S}$ has exactly one non-zero entry per column. Further, $\mathbf{W} = \mathbf{S}^\top\mathbf{KS} = (\mathbf{XS})^\top\mathbf{XS} = \mathbf{X}'^\top\mathbf{X}'$, where $\mathbf{X}' \in \mathbb{R}^{N\times l}$ contains l sampled columns of $\mathbf{X}$ and $\mathbf{X}' = \mathbf{U}_{X'}\mathbf{\Sigma}_{X'}\mathbf{V}_{X'}^\top$ is the SVD of $\mathbf{X}'$. We use these definitions to present Theorems 1 and 2.

Theorem 1 *Column sampling and Nyström spectral reconstructions of rank k are of the form*

$$\mathbf{X}^\top\mathbf{U}_{X',k}\mathbf{Z}\mathbf{U}_{X',k}^\top\mathbf{X},$$

where $\mathbf{Z} \in \mathbb{R}^{k\times k}$ is SPSD. Further, among all approximations of this form, neither the Column sampling nor the Nyström approximation is optimal (in $\|\cdot\|_F$).

Proof. If $\alpha = \sqrt{n/l}$, then starting from (6.8) and expressing $\mathbf{C}$ and $\mathbf{W}$ in terms of $\mathbf{X}$ and $\mathbf{S}$, we have

$$\begin{aligned}\widetilde{\mathbf{K}}_k^{col} &= \alpha\mathbf{KS}((\mathbf{S}^\top\mathbf{K}^2\mathbf{S})_k^{1/2})^+\mathbf{S}^\top\mathbf{K}^\top \\ &= \alpha\mathbf{X}^\top\mathbf{X}'\big((\mathbf{V}_{C,k}\mathbf{\Sigma}_{C,k}^2\mathbf{V}_{C,k}^\top)^{1/2}\big)^+\mathbf{X}'^\top\mathbf{X} \\ &= \mathbf{X}^\top\mathbf{U}_{X',k}\mathbf{Z}_{col}\mathbf{U}_{X',k}^\top\mathbf{X}, \end{aligned} \tag{6.9}$$

where $\mathbf{Z}_{col} = \alpha \mathbf{\Sigma}_{X'} \mathbf{V}_{X'}^\top \mathbf{V}_{C,k} \mathbf{\Sigma}_{C,k}^{+} \mathbf{V}_{C,k}^\top \mathbf{V}_{X'} \mathbf{\Sigma}_{X'}$. Similarly, from (6.6) we have:

$$\begin{aligned}\widetilde{\mathbf{K}}_k^{nys} =& \mathbf{K}\mathbf{S}(\mathbf{S}^\top\mathbf{K}\mathbf{S})_k^{+}\mathbf{S}^\top\mathbf{K}^\top \\ =& \mathbf{X}^\top\mathbf{X}'\big(\mathbf{X}'^\top\mathbf{X}'\big)_k^{+}\mathbf{X}'^\top\mathbf{X} \\ =& \mathbf{X}^\top\mathbf{U}_{X',k}\mathbf{U}_{X',k}^\top\mathbf{X}. \end{aligned} \tag{6.10}$$

Clearly, $\mathbf{Z}_{nys} = \mathbf{I}_k$. Next, we analyze the error, $\mathbf{E}$, for an arbitrary $\mathbf{Z}$, which yields the approximation $\widetilde{\mathbf{K}}_k^Z$:

$$\mathbf{E} = \|\mathbf{K} - \widetilde{\mathbf{K}}_k^Z\|_F^2 = \|\mathbf{X}^\top(\mathbf{I}_N - \mathbf{U}_{X',k}\mathbf{Z}\mathbf{U}_{X',k}^\top)\mathbf{X}\|_F^2. \tag{6.11}$$

Let $\mathbf{X} = \mathbf{U}_X\mathbf{\Sigma}_X\mathbf{V}_X^\top$ and $\mathbf{Y} = \mathbf{U}_X^\top\mathbf{U}_{X',k}$. Then,

$$\begin{aligned}\mathbf{E} &= \mathrm{Tr}\left[\big((\mathbf{I}_N - \mathbf{U}_{X',k}\mathbf{Z}\mathbf{U}_{X',k}^\top)\mathbf{U}_X\mathbf{\Sigma}_X^2\mathbf{U}_X^\top\big)^2\right] \\ &= \mathrm{Tr}\left[\big(\mathbf{U}_X\mathbf{\Sigma}_X\mathbf{U}_X^\top(\mathbf{I}_N - \mathbf{U}_{X',k}\mathbf{Z}\mathbf{U}_{X',k}^\top)\mathbf{U}_X\mathbf{\Sigma}_X\mathbf{U}_X^\top\big)^2\right] \\ &= \mathrm{Tr}\left[\big(\mathbf{U}_X\mathbf{\Sigma}_X(\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top)\mathbf{\Sigma}_X\mathbf{U}_X^\top\big)^2\right] \\ &= \mathrm{Tr}\left[\mathbf{\Sigma}_X(\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top)\mathbf{\Sigma}_X^2(\mathbf{I}_N - \mathbf{Y}\mathbf{Z}\mathbf{Y}^\top)\mathbf{\Sigma}_X)\right] \\ &= \mathrm{Tr}\left[\mathbf{\Sigma}_X^4 - 2\mathbf{\Sigma}_X^2\mathbf{Y}\mathbf{Z}\mathbf{Y}^\top\mathbf{\Sigma}_X^2 + \mathbf{\Sigma}_X\mathbf{Y}\mathbf{Z}\mathbf{Y}^\top\mathbf{\Sigma}_X^2\mathbf{Y}\mathbf{Z}\mathbf{Y}^\top\mathbf{\Sigma}_X)\right]. \end{aligned} \tag{6.12}$$

To find $\mathbf{Z}^*$, the $\mathbf{Z}$ that minimizes (6.12), we use the convexity of (6.12) and set:

$$\partial\mathbf{E}/\partial\mathbf{Z} = -2\mathbf{Y}^\top\mathbf{\Sigma}_X^4\mathbf{Y} + 2(\mathbf{Y}^\top\mathbf{\Sigma}_X^2\mathbf{Y})\mathbf{Z}^*(\mathbf{Y}^\top\mathbf{\Sigma}_X^2\mathbf{Y}) = 0$$

and solve for $\mathbf{Z}^*$, which gives us:

$$\mathbf{Z}^* = (\mathbf{Y}^\top\mathbf{\Sigma}_X^2\mathbf{Y})^+(\mathbf{Y}^\top\mathbf{\Sigma}_X^4\mathbf{Y})(\mathbf{Y}^\top\mathbf{\Sigma}_X^2\mathbf{Y})^+.$$

$\mathbf{Z}^* = \mathbf{Z}_{nys} = \mathbf{I}_k$ if $\mathbf{Y} = \mathbf{I}_k$, though $\mathbf{Z}^*$ does not in general equal either $\mathbf{Z}_{col}$ or $\mathbf{Z}_{nys}$, which is clear by comparing the expressions of these three matrices.[6] Furthermore, since $\mathbf{\Sigma}_X^2 = \mathbf{\Sigma}_K$, $\mathbf{Z}^*$ depends on the spectrum of $\mathbf{K}$. $\square$

While Theorem 1 shows that the optimal approximation is data dependent and may differ from the Nyström and Column sampling approximations, Theorem 2 presented below reveals that in certain instances the Nyström method is optimal. In contrast, the Column sampling method enjoys no such guarantee.

Theorem 2 *Let* $r = \mathrm{rank}(\mathbf{K}) \leq k \leq l$ *and* $\mathrm{rank}(\mathbf{W}) = r$. *Then, the Nyström approximation is exact for spectral reconstruction. In contrast, Column sampling is exact iff* $\mathbf{W} = \big((l/n)\mathbf{C}^\top\mathbf{C}\big)^{1/2}$. *When this specific condition holds, Column-Sampling trivially reduces to the Nyström method.*

Proof. Since $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$, $\mathrm{rank}(\mathbf{K}) = \mathrm{rank}(\mathbf{X}) = r$. Similarly, $\mathbf{W} = \mathbf{X}'^\top\mathbf{X}'$ implies $\mathrm{rank}(\mathbf{X}') = r$. Thus the columns of $\mathbf{X}'$ span the columns of $\mathbf{X}$ and $\mathbf{U}_{X',r}$

[6]This fact is illustrated in our experimental results for the 'DEXT' dataset in Figure 6.2(a).

Table 6.2: Description of the datasets used in our experiments comparing sampling-based matrix approximations (Sim et al., 2002; LeCun & Cortes, 1998; Talwalkar et al., 2008). 'n' denotes the number of points and 'd' denotes the data dimensionality, i.e., the number of features in input space.

Dataset	Data	n	d	Kernel
PIE-2.7K	faces	2731	2304	linear
PIE-7K	faces	7412	2304	linear
MNIST	digits	4000	784	linear
ESS	proteins	4728	16	RBF
ABN	abalones	4177	8	RBF

is an orthonormal basis for $\mathbf{X}$, i.e., $\mathbf{I}_N - \mathbf{U}_{X',r}\mathbf{U}_{X',r}^\top \in \text{Null}(\mathbf{X})$. Since $k \geq r$, from (6.10) we have

$$\|\mathbf{K} - \widetilde{\mathbf{K}}_k^{nys}\|_F = \|\mathbf{X}^\top(\mathbf{I}_N - \mathbf{U}_{X',r}\mathbf{U}_{X',r}^\top)\mathbf{X}\|_F = 0, \tag{6.13}$$

which proves the first statement of the theorem. To prove the second statement, we note that $\text{rank}(C) = r$. Thus, $\mathbf{C} = \mathbf{U}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top$ and $(\mathbf{C}^\top\mathbf{C})_k^{1/2} = (\mathbf{C}^\top\mathbf{C})^{1/2} = \mathbf{V}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top$ since $k \geq r$. If $\mathbf{W} = (1/\alpha)(\mathbf{C}^\top\mathbf{C})^{1/2}$, then the Column sampling and Nyström approximations are identical and hence exact. Conversely, to exactly reconstruct $\mathbf{K}$, Column sampling necessarily reconstructs $\mathbf{C}$ exactly. Using $\mathbf{C}^\top = [\mathbf{W} \quad \mathbf{K}_{21}^\top]$ in (6.8) we have:

$$\begin{aligned}
\widetilde{\mathbf{K}}_k^{col} = \mathbf{K} \quad &\Longrightarrow \quad \alpha\mathbf{C}\big((\mathbf{C}^\top\mathbf{C})_k^{\frac{1}{2}}\big)^+\mathbf{W} = \mathbf{C} && (6.14)\\
&\Longrightarrow \quad \alpha\mathbf{U}_{C,r}\mathbf{V}_{C,r}^\top\mathbf{W} = \mathbf{U}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top && (6.15)\\
&\Longrightarrow \quad \alpha\mathbf{V}_{C,r}\mathbf{V}_{C,r}^\top\mathbf{W} = \mathbf{V}_{C,r}\mathbf{\Sigma}_{C,r}\mathbf{V}_{C,r}^\top && (6.16)\\
&\Longrightarrow \quad \mathbf{W} = \frac{1}{\alpha}(\mathbf{C}^\top\mathbf{C})^{1/2}. && (6.17)
\end{aligned}$$

In (6.16) we use $\mathbf{U}_{C,r}^\top\mathbf{U}_{C,r} = \mathbf{I}_r$, while (6.17) follows since $\mathbf{V}_{C,r}\mathbf{V}_{C,r}^\top$ is an orthogonal projection onto the span of the rows of $\mathbf{C}$ and the columns of $\mathbf{W}$ lie within this span implying $\mathbf{V}_{C,r}\mathbf{V}_{C,r}^\top\mathbf{W} = \mathbf{W}$. $\square$

6.3.3 Experiments

To test the accuracy of singular values/vectors and low-rank approximations for different methods, we used several kernel matrices arising in different applications, as described in Table 6.2. We worked with datasets containing less than ten thousand points to be able to compare with exact SVD. We fixed k to be 100 in all the experiments, which captures more than 90% of the spectral energy for each dataset.

For singular values, we measured percentage accuracy of the approximate singular values with respect to the exact ones. For a fixed l, we performed 10 trials by selecting columns uniformly at random from $\mathbf{K}$. We show in Figure 6.1(a) the difference in mean percentage accuracy for the two methods for $l = n/10$, with results bucketed by groups of singular values, i.e., we sorted the singular values in descending order, grouped them as indicated in the figure, and report the average percentage accuracy for each group. The empirical results show that the Column sampling method generates more accurate singular values than the Nyström method. A similar trend was observed for other values of l.

For singular vectors, the accuracy was measured by the dot product i.e., cosine of principal angles between the exact and the approximate singular vectors. Figure 6.1(b) shows the difference in mean accuracy between Nyström and Column sampling methods, once again bucketed by groups of singular vectors sorted in descending order based on their corresponding singular values. The top 100 singular vectors were all better approximated by Column sampling for all datasets. This trend was observed for other values of l as well. Furthermore, even when the Nyström singular vectors are orthogonalized, the Column sampling approximations are superior, as shown in Figure 6.1(c).

Next we compared the low-rank approximations generated by the two methods using spectral reconstruction as described in Section 6.3.2. We measured the accuracy of reconstruction relative to the optimal rank-k approximation, $\mathbf{K}_k$, as:

$$\text{relative accuracy} = \frac{\|\mathbf{K} - \mathbf{K}_k\|_F}{\|\mathbf{K} - \widetilde{\mathbf{K}}_k^{nys/col}\|_F}. \tag{6.18}$$

The relative accuracy will approach one for good approximations. Results are shown in Figure 6.2(a). The Nyström method produces superior results for spectral reconstruction. These results are somewhat surprising given the relatively poor quality of the singular values/vectors for the Nyström method, but they are in agreement with the consequences of Theorem 2. Furthermore, as stated in Theorem 1, the optimal spectral reconstruction approximation is tied to the spectrum of $\mathbf{K}$. Our results suggest that the relative accuracies of Nyström and Column sampling spectral reconstructions are also tied to this spectrum. When we analyzed spectral reconstruction performance on a sparse kernel matrix with a slowly decaying spectrum, we found that Nyström and Column sampling approximations were roughly equivalent ('DEXT' in Figure 6.2(a)). This result contrasts the results for dense kernel matrices with exponentially decaying spectra arising from the other datasets used in the experiments.

One factor that impacts the accuracy of the Nyström method for some tasks is the non-orthonormality of its singular vectors (Section 6.3.1). Although orthonormalization is computationally costly and typically avoided in practice, we nonetheless evaluated the effect of such orthonormalization. Empirically, the accuracy of Orthonormal Nyström spectral reconstruction is actually worse relative to the standard Nyström approximation, as shown in Figure 6.2(b). This surprising result can be attributed to the fact that orthonormalization of the singular vectors leads to the loss of some of the unique properties described in Section 6.3.2. For instance, Theorem 2 no longer holds and the scaling terms do not cancel out, i.e., $\widetilde{\mathbf{K}}_k^{nys} \neq \mathbf{C}\mathbf{W}_k^{+}\mathbf{C}^{\top}$.

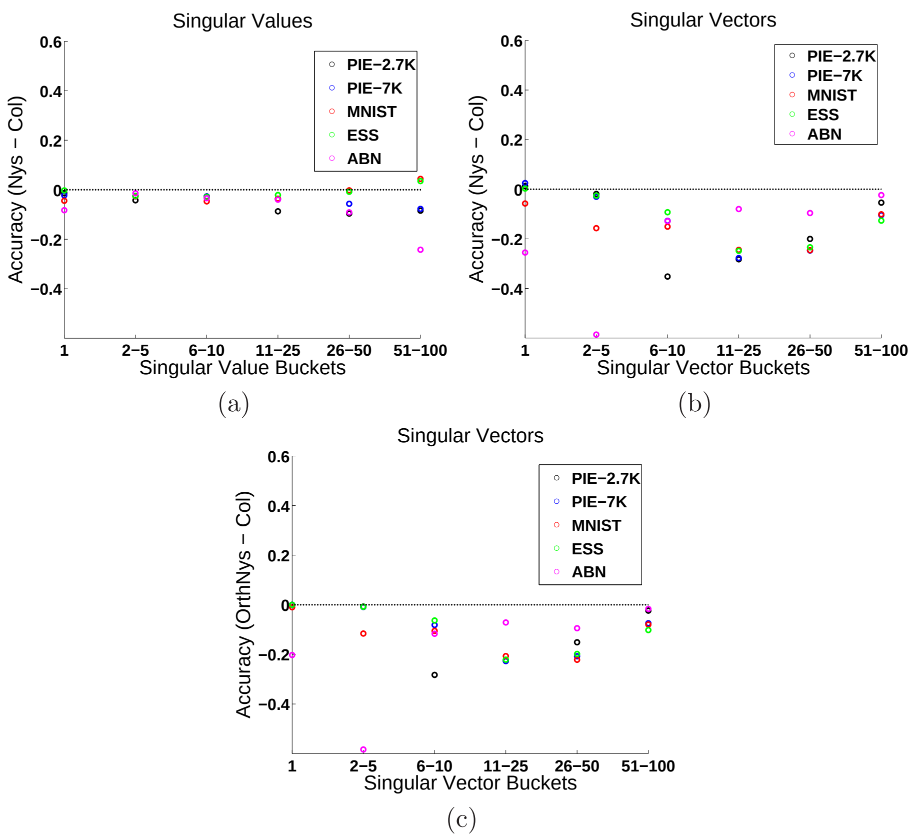

Figure 6.1: Differences in accuracy between Nyström and Column-Sampling. Values above zero indicate better performance of Nyström and vice-versa. (a) Top 100 singular values with $l = n/10$. (b) Top 100 singular vectors with $l = n/10$. (c) Comparison using orthogonalized Nyström singular vectors.

6.4 Large-scale Manifold Learning

In the previous section, we discussed two sampling-based techniques that generate approximations for kernel matrices. Although we analyzed the effectiveness of these techniques for approximating singular values, singular vectors and low-rank matrix reconstruction, we have yet to discuss the effectiveness of these techniques in the context of actual machine learning tasks. In fact, the Nyström method has been shown to be successful on a variety of learning tasks including Support Vector Machines (Fine & Scheinberg, 2002), Gaussian Processes (Williams & Seeger, 2000), Spectral Clustering (Fowlkes et al., 2004), manifold learning (Talwalkar et al., 2008), Kernel Logistic Regression (Karsmakers et al., 2007), Kernel Ridge Regression (Cortes et al., 2010) and more generally

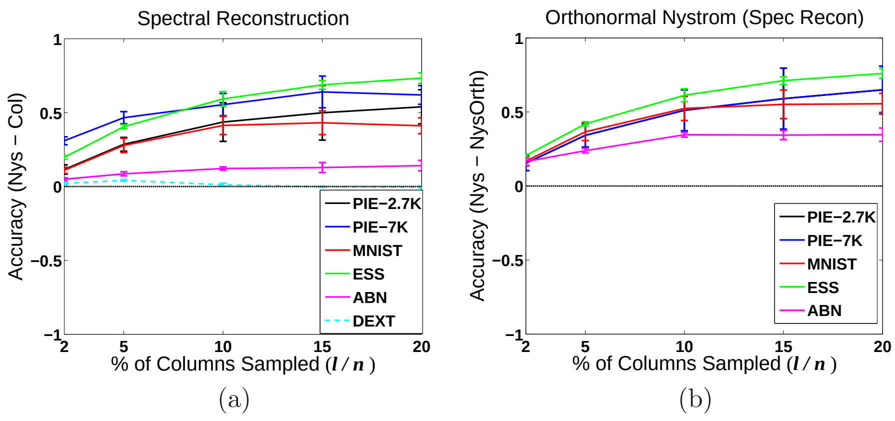

Figure 6.2: Performance accuracy of spectral reconstruction approximations for different methods with $k = 100$. Values above zero indicate better performance of the Nyström method. (a) Nyström versus Column sampling. (b) Nyström versus Orthonormal Nyström.

to approximate regularized matrix inverses via the Woodbury approximation (Williams & Seeger, 2000). In this section, we will discuss in detail how approximate embeddings can be used in the context of manifold learning, relying on the sampling based algorithms from the previous section to generate an approximate SVD. In particular, we present the largest study to date for manifold learning, and provide a quantitative comparison of Isomap and Laplacian Eigenmaps for large scale face manifold construction on clustering and classification tasks.

6.4.1 Manifold learning

Manifold learning considers the problem of extracting low-dimensional structure from high-dimensional data. Given n input points, $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ and $\mathbf{x}_i \in \mathbb{R}^d$, the goal is to find corresponding outputs $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i \in \mathbb{R}^k$, $k \ll d$, such that $\mathbf{Y}$ 'faithfully' represents $\mathbf{X}$. We now briefly review the Isomap and Laplacian Eigenmaps techniques to discuss their computational complexity.

Isomap

Isomap aims to extract a low-dimensional data representation that best preserves all pairwise distances between input points, as measured by their geodesic distances along the manifold (Tenenbaum et al., 2000). It approximates the geodesic distance assuming that input space distance provides good approximations for nearby points, and for faraway points it estimates distance as a series of hops between neighboring points. This approximation becomes exact in the limit of infinite data. Isomap can be viewed as an adaptation of Classical

Multidimensional Scaling (Cox et al., 2000), in which geodesic distances replace Euclidean distances.

Computationally, Isomap requires three steps:

1. Find the t nearest neighbors for each point in input space and construct an undirected neighborhood graph, denoted by $\mathcal{G}$, with points as nodes and links between neighbors as edges. This requires $\mathrm{O}(n^2)$ time.

2. Compute the approximate geodesic distances, Δ_{ij}, between all pairs of nodes (i, j) by finding shortest paths in $\mathcal{G}$ using Dijkstra's algorithm at each node. Perform double centering, which converts the squared distance matrix into a dense $n \times n$ similarity matrix, i.e., compute $\mathbf{K} = -\frac{1}{2}\mathbf{HDH}$, where $\mathbf{D}$ is the squared distance matrix, $\mathbf{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the centering matrix, $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\mathbf{1}$ is a column vector of all ones. This step takes $\mathrm{O}(n^2 \log n)$ time, dominated by the calculation of geodesic distances.

3. Find the optimal k dimensional representation, $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n$, such that $\mathbf{Y} = \mathrm{argmin}_{\mathbf{Y}'} \sum_{i,j} \left(\|\mathbf{y}'_i - \mathbf{y}'_j\|_2^2 - \Delta_{ij}^2\right)$. The solution is given by,

$$\mathbf{Y} = (\mathbf{\Sigma}_k)^{1/2}\mathbf{U}_k^\top \tag{6.19}$$

   where $\mathbf{\Sigma}_k$ is the diagonal matrix of the top k singular values of $\mathbf{K}$ and $\mathbf{U}_k$ are the associated singular vectors. This step requires $\mathrm{O}(n^2)$ space for storing $\mathbf{K}$, and $\mathrm{O}(n^3)$ time for its SVD.

The time and space complexities for all three steps are intractable for $n = 18\mathrm{M}$.

Laplacian Eigenmaps

Laplacian Eigenmaps aims to find a low-dimensional representation that best preserves neighborhood relations as measured by a weight matrix $\mathbf{W}$ (Belkin & Niyogi, 2001).[7] The algorithm works as follows:

1. Similar to Isomap, first find t nearest neighbors for each point. Then construct $\mathbf{W}$, a sparse, symmetric $n \times n$ matrix, where $\mathbf{W}_{ij} = \exp\big(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma^2\big)$ if $(\mathbf{x}_i, \mathbf{x}_j)$ are neighbors, 0 otherwise, and σ is a scaling parameter.

2. Construct the diagonal matrix $\mathbf{D}$, such that $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$, in $\mathrm{O}(tn)$ time.

3. Find the k dimensional representation by minimizing the normalized, weighted distance between neighbors as,

$$\mathbf{Y} = \underset{\mathbf{Y}'}{\mathrm{argmin}} \sum_{i,j} \left(\frac{\mathbf{W}_{ij}\|\mathbf{y}'_i - \mathbf{y}'_j\|_2^2}{\sqrt{\mathbf{D}_{ii}\mathbf{D}_{jj}}} \right). \tag{6.20}$$

[7] The weight matrix should not be confused with the subsampled SPSD matrix, $\mathbf{W}$, associated with the Nyström method. Since sampling-based approximation techniques will not be used with Laplacian Eigenmaps, the notation should be clear from the context.

This objective function penalizes nearby inputs for being mapped to faraway outputs, with 'nearness' measured by the weight matrix $\mathbf{W}$ (Chapelle et al., 2006). To find $\mathbf{Y}$, we define $\mathcal{L} = \mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ where $\mathcal{L} \in \mathbb{R}^{n\times n}$ is the symmetrized, normalized form of the graph Laplacian, given by $\mathbf{D} - \mathbf{W}$. Then, the solution to the minimization in (6.20) is

$$\mathbf{Y} = \mathbf{U}_{\mathcal{L},k}^{\top} \tag{6.21}$$

where $\mathbf{U}_{\mathcal{L},k}^{\top}$ are the bottom k singular vectors of $\mathcal{L}$, excluding the last singular vector corresponding to the singular value 0. Since $\mathcal{L}$ is sparse, it can be stored in $\mathrm{O}(tn)$ space, and iterative methods, such as Lanczos, can be used to find these k singular vectors relatively quickly.

To summarize, in both the Isomap and Laplacian Eigenmaps methods, the two main computational efforts required are neighborhood graph construction and manipulation and SVD of a symmetric positive semidefinite (SPSD) matrix. In the next section, we further discuss the Nyström and Column sampling methods in the context of manifold learning, and describe the graph operations in Section 6.4.3.

6.4.2 Approximation experiments

Since we use sampling-based SVD approximation to scale Isomap, we first examined how well the Nyström and Column sampling methods approximated our desired low-dimensional embeddings, i.e., $\mathbf{Y} = (\mathbf{\Sigma}_k)^{1/2}\mathbf{U}_k^{\top}$. Using (6.3), the Nyström low-dimensional embeddings are:

$$\widetilde{\mathbf{Y}}^{nys} = \widetilde{\mathbf{\Sigma}}_{nys,k}^{1/2}\widetilde{\mathbf{U}}_{nys,k}^{\top} = \big((\mathbf{\Sigma}_W)_k^{1/2}\big)^{+}\mathbf{U}_{W,k}^{\top}\mathbf{C}^{\top}. \tag{6.22}$$

Similarly, from (6.4) we can express the Column sampling low-dimensional embeddings as:

$$\widetilde{\mathbf{Y}}^{col} = \widetilde{\mathbf{\Sigma}}_{col,k}^{1/2}\widetilde{\mathbf{U}}_{col,k}^{\top} = \sqrt[4]{\frac{n}{l}}\big((\mathbf{\Sigma}_C)_k^{1/2}\big)^{+}\mathbf{V}_{C,k}^{\top}\mathbf{C}^{\top}. \tag{6.23}$$

Both approximations are of a similar form. Further, notice that the optimal low-dimensional embeddings are in fact the square root of the optimal rank k approximation to the associated SPSD matrix, i.e., $\mathbf{Y}^{\top}\mathbf{Y} = \mathbf{K}_k$, for Isomap. As such, there is a connection between the task of approximating low-dimensional embeddings and the task of generating low-rank approximate spectral reconstructions, as discussed in Section 6.3.2. Recall that the theoretical analysis in Section 6.3.2 as well as the empirical results in Section 6.3.3 both suggested that the Nyström method was superior in its spectral reconstruction accuracy. Hence, we performed an empirical study using the datasets from Table 6.2 to measure the quality of the low-dimensional embeddings generated by the two techniques and see if the same trend exists.

We measured the quality of the low-dimensional embeddings by calculating the extent to which they preserve distances, which is the appropriate criterion

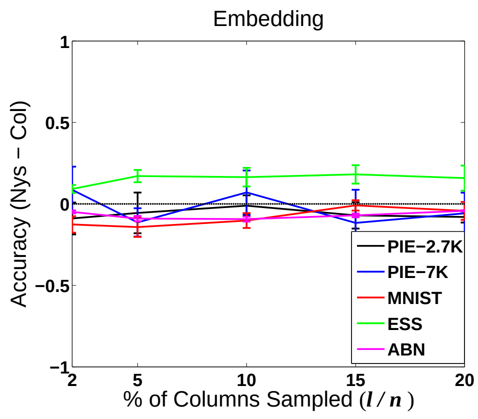

Figure 6.3: Embedding accuracy of Nyström and Column-Sampling. Values above zero indicate better performance of Nyström and vice-versa.

in the context of manifold learning. For each dataset, we started with a kernel matrix, $\mathbf{K}$, from which we computed the associated $n \times n$ squared distance matrix, $\mathbf{D}$, using the fact that $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij}$. We then computed the approximate low-dimensional embeddings using the Nyström and Column sampling methods, and then used these embeddings to compute the associated approximate squared distance matrix, $\widetilde{\mathbf{D}}$. We measured accuracy using the notion of relative accuracy defined in (6.18), which can be expressed in terms of distance matrices as:

$$\text{relative accuracy} = \frac{\|\mathbf{D} - \mathbf{D}_k\|_F}{\|\mathbf{D} - \widetilde{\mathbf{D}}\|_F},$$

where $\mathbf{D}_k$ corresponds to the distance matrix computed from the optimal k dimensional embeddings obtained using the singular values and singular vectors of $\mathbf{K}$. In our experiments, we set $k = 100$ and used various numbers of sampled columns, ranging from $l = n/50$ to $l = n/5$. Figure 6.3 presents the results of our experiments. Surprisingly, we do not see the same trend in our empirical results for embeddings as we previously observed for spectral reconstruction, as the two techniques exhibit roughly similar behavior across datasets. As a result, we decided to use both the Nyström and Column sampling methods for our subsequent manifold learning study.

6.4.3 Large-scale learning

In this section, we outline the process of learning a manifold of faces. We first describe the datasets used in our experiments. We then explain how to extract nearest neighbors, a common step between Laplacian Eigenmaps and Isomap. The remaining steps of Laplacian Eigenmaps are straightforward, so the subsequent sections focus on Isomap, and specifically on the computational efforts required to generate a manifold using Webfaces-18M.

Datasets

We used two faces datasets consisting of 35K and 18M images. The CMU PIE face dataset (Sim et al., 2002) contains $41,368$ images of 68 subjects under 13 different poses and various illumination conditions. A standard face detector extracted $35,247$ faces (each 48×48 pixels), which comprised our 35K set (PIE-35K). We used this set because, being labeled, it allowed us to perform quantitative comparisons. The second dataset, named Webfaces-18M, contains 18.2 million images of faces extracted from the Web using the same face detector. For both datasets, face images were represented as 2304 dimensional pixel vectors which were globally normalized to have zero mean and unit variance. No other pre-processing, e.g., face alignment, was performed. In contrast, He et al. (2005) used well-aligned faces (as well as much smaller data sets) to learn face manifolds. Constructing Webfaces-18M, including face detection and duplicate removal, took 15 hours using a cluster of several hundred machines. We used this cluster for all experiments requiring distributed processing and data storage.

Nearest neighbors and neighborhood graph

The cost of naive nearest neighbor computation is $\mathrm{O}(n^2)$, where n is the size of the dataset. It is possible to compute exact neighbors for PIE-35K, but for Webfaces-18M this computation is prohibitively expensive. So, for this set, we used a combination of random projections and spill trees (Liu et al., 2004) to get approximate neighbors. Computing 5 nearest neighbors in parallel with spill trees took $\sim$2 days on the cluster. Figure 6.4 shows the top 5 neighbors for a few randomly chosen images in Webfaces-18M. In addition to this visualization, comparison of exact neighbors and spill tree approximations for smaller subsets suggested good performance of spill trees.

We next constructed the neighborhood graph by representing each image as a node and connecting all neighboring nodes. Since Isomap and Laplacian Eigenmaps require this graph to be connected, we used depth-first search to find its largest connected component. These steps required $\mathrm{O}(tn)$ space and time. Constructing the neighborhood graph for Webfaces-18M and finding the largest connected component took 10 minutes on a single machine using the OpenFST library (Allauzen et al., 2007).

For neighborhood graph construction, an 'appropriate' choice of number of neighbors, t, is crucial. A small t may give too many disconnected components, while a large t may introduce unwanted edges. These edges stem from inadequately sampled regions of the manifold and false positives introduced by the face detector. Since Isomap needs to compute shortest paths in the neighborhood graph, the presence of bad edges can adversely impact these computations. This is known as the problem of leakage or 'short-circuits' (Balasubramanian & Schwartz, 2002). Here, we chose $t = 5$ and also enforced an upper limit on neighbor distance to alleviate the problem of leakage. We used a distance limit corresponding to the 95^{th} percentile of neighbor distances in the PIE-35K

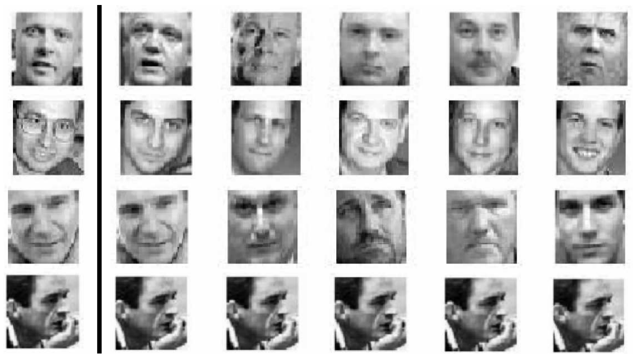

Figure 6.4: Visualization of neighbors for Webfaces-18M. The first image in each row is the input, and the next five are its neighbors.

	No Upper Limit		Upper Limit Enforced	
t	# Comp	% Largest	# Comp	% Largest
1	1.7M	0.05 %	4.3M	0.03 %
2	97K	97.2 %	285K	80.1 %
3	18K	99.3 %	277K	82.2 %
5	1.9K	99.9 %	275K	83.1 %

Table 6.3: Number of components in the Webfaces-18M neighbor graph and the percentage of images within the largest connected component ('% Largest') for varying numbers of neighbors (t) with and without an upper limit on neighbor distances.

dataset.

Table 6.3 shows the effect of choosing different values for t with and without enforcing the upper distance limit. As expected, the size of the largest connected component increases as t increases. Also, enforcing the distance limit reduces the size of the largest component. Figure 6.5 shows a few random samples from the largest component. Images not within the largest component are either part of a strongly connected set of images (Figure 6.6) or do not have any neighbors within the upper distance limit (Figure 6.7). There are significantly more false positives in Figure 6.7 than in Figure 6.5, although some of the images in Figure 6.7 are actually faces. Clearly, the distance limit introduces a trade-off between filtering out non-faces and excluding actual faces from the largest component.[8]

[8]To construct embeddings with Laplacian Eigenmaps, we generated **W** and **D** from nearest neighbor data for images within the largest component of the neighborhood graph and solved (6.21) using a sparse eigensolver.

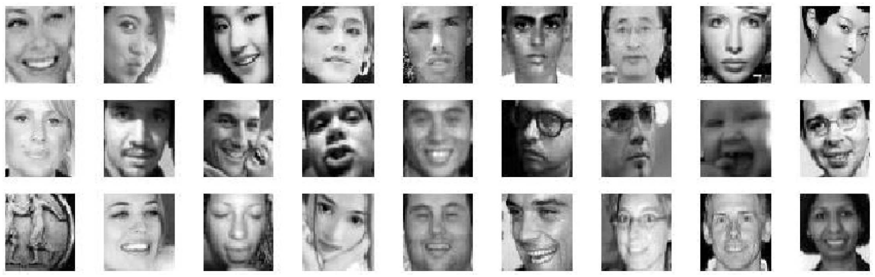

Figure 6.5: A few random samples from the largest connected component of the Webfaces-18M neighborhood graph.

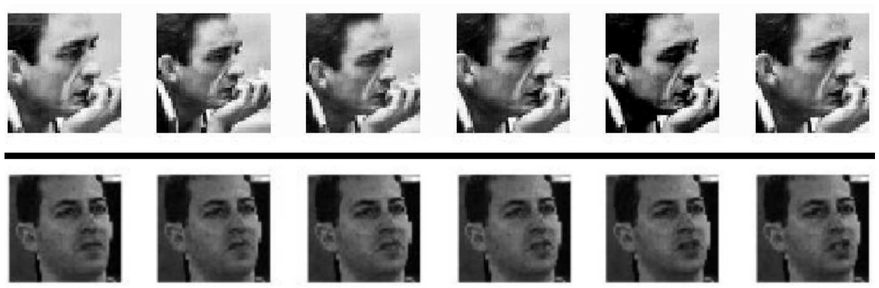

Figure 6.6: Visualization of disconnected components of the neighborhood graphs from Webfaces-18M (top row) and from PIE-35K (bottom row). The neighbors for each of these images are all within this set, thus making the entire set disconnected from the rest of the graph. Note that these images are not exactly the same.

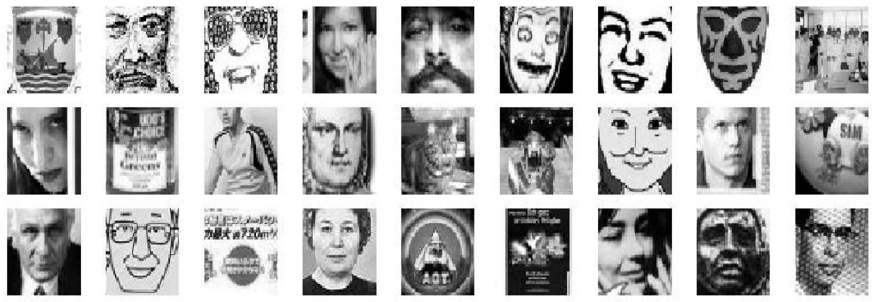

Figure 6.7: Visualization of disconnected components containing exactly one image. Although several of the images above are not faces, some are actual faces, suggesting that certain areas of the face manifold are not adequately sampled by Webfaces-18M.

Approximating geodesics

To construct the similarity matrix $\mathbf{K}$ in Isomap, one approximates geodesic distance by shortest-path lengths between every pair of nodes in the neighborhood graph. This requires $O(n^2 \log n)$ time and $O(n^2)$ space, both of which are prohibitive for 18M nodes. However, since we use sampling-based approximate decomposition, we need only $l \ll n$ columns of $\mathbf{K}$, which form the submatrix $\mathbf{C}$. We thus computed geodesic distance between l randomly selected nodes (called landmark points) and the rest of the nodes, which required $O(ln \log n)$ time and $O(ln)$ space. Since this computation can easily be parallelized, we performed geodesic computation on the cluster and stored the output in a distributed fashion. The overall procedure took 60 minutes for Webfaces-18M using $l = 10$K. The bottom four rows in Figure 6.9 show sample shortest paths for images within the largest component for Webfaces-18M, illustrating smooth transitions between images along each path.[9]

Generating low-dimensional embeddings

Before generating low-dimensional embeddings using Isomap, one needs to convert distances into similarities using a process called centering (Cox et al., 2000). For the Nyström approximation, we computed $\mathbf{W}$ by double centering $\mathbf{D}$, the $l \times l$ matrix of squared geodesic distances between all landmark nodes, as $\mathbf{W} = -\frac{1}{2}\mathbf{HDH}$, where $\mathbf{H} = \mathbf{I}_l - \frac{1}{l}\mathbf{1}\mathbf{1}^\top$ is the centering matrix, $\mathbf{I}_l$ is the $l \times l$ identity matrix and $\mathbf{1}$ is a column vector of all ones. Similarly, the matrix $\mathbf{C}$ was obtained from squared geodesic distances between the landmark nodes and all other nodes using single-centering as described in de Silva and Tenenbaum (2003).

For the Column sampling approximation, we decomposed $\mathbf{C}^\top\mathbf{C}$, which we constructed by performing matrix multiplication in parallel on $\mathbf{C}$. For both approximations, decomposition on an $l \times l$ matrix ($\mathbf{C}^\top\mathbf{C}$ or $\mathbf{W}$) took about one hour. Finally, we computed low-dimensional embeddings by multiplying the scaled singular vectors from approximate decomposition with $\mathbf{C}$. For Webfaces-18M, generating low dimensional embeddings took 1.5 hours for the Nyström method and 6 hours for the Column sampling method.

6.4.4 Manifold evaluation

Manifold learning techniques typically transform the data such that Euclidean distance in the transformed space between *any* pair of points is meaningful, under the assumption that in the original space Euclidean distance is meaningful only in local neighborhoods. Since K-means clustering computes Euclidean distances between all pairs of points, it is a natural choice for evaluating these techniques. We also compared the performance of various techniques using

[9]In fact, the techniques we described in the context of approximating geodesic distances via shortest path are currently used by Google for its "People Hopper" application which runs on the social networking site Orkut (Kumar & Rowley, 2010).

nearest neighbor classification. Since CMU-PIE is a labeled dataset, we first focused on quantitative evaluation of different embeddings using face pose as class labels. The PIE set contains faces in 13 poses, and such a fine sampling of the pose space makes clustering and classification tasks very challenging. In all the experiments we fixed the dimension of the reduced space, k, to be 100.

The first set of experiments was aimed at finding how well different Isomap approximations perform in comparison to exact Isomap. We used a subset of PIE with 10K images (PIE-10K) since, for this size, exact SVD could be done on a single machine within reasonable time and memory limits. We fixed the number of clusters in our experiments to equal the number of pose classes, and measured clustering performance using two measures, *Purity* and *Accuracy*. Purity measures the frequency of data belonging to the same cluster sharing the same class label, while Accuracy measures the frequency of data from the same class appearing in a single cluster. Thus, ideal clustering will have 100% Purity and 100% Accuracy.

Table 6.4 shows that clustering with Nyström Isomap with just $l = 1$K performs almost as well as exact Isomap on this dataset[10]. This matches with the observation made in Williams and Seeger (2000), where the Nyström approximation was used to speed up kernel machines. Also, Column sampling Isomap performs slightly worse than Nyström Isomap. The clustering results on the full PIE-35K set (Table 6.5) with $l = 10$K also affirm this observation. Figure 6.8 shows the optimal 2D projections from different methods for PIE-35K. The Nyström method separates the pose clusters better than Column sampling verifying the quantitative results.

The fact that Nyström outperforms Column sampling is somewhat surprising given the experimental evaluations in Section 6.4.2, where we found the two approximation techniques to achieve similar performance. One possible reason for the poor performance of Column sampling Isomap is due to the form of the similarity matrix $\mathbf{K}$. When using a finite number of data points for Isomap, $\mathbf{K}$ is not guaranteed to be SPSD. We verified that $\mathbf{K}$ was not SPSD in our experiments, and a significant number of top eigenvalues, i.e., those with largest magnitudes, were negative. The two approximation techniques differ in their treatment of negative eigenvalues and the corresponding eigenvectors. The Nyström method allows one to use eigenvalue decomposition (EVD) of $\mathbf{W}$ to yield signed eigenvalues, making it possible to discard the negative eigenvalues and the corresponding eigenvectors. On the contrary, it is not possible to discard these in the Column-based method, since the signs of eigenvalues are lost in the SVD of the rectangular matrix $\mathbf{C}$ (or EVD of $\mathbf{C}^\top\mathbf{C}$). Thus, the presence of negative eigenvalues deteriorates the performance of Column sampling method more than the Nyström method.

Tables 6.4 and 6.5 also show a significant difference in the Isomap and Laplacian Eigenmaps results. The 2D embeddings of PIE-35K (Figure 6.8) reveal that Laplacian Eigenmaps projects data points into a small compact region, consistent with its objective function defined in (6.20), as it tends to map neighboring

[10]The differences are statistically insignificant.

Methods	Purity (%)	Accuracy (%)
PCA	54.3 ($\pm$0.8)	46.1 ($\pm$1.4)
Exact Isomap	58.4 ($\pm$1.1)	53.3 ($\pm$4.3)
Nyström Isomap	59.1 ($\pm$0.9)	53.3 ($\pm$2.7)
Col-Sampling Isomap	56.5 ($\pm$0.7)	49.4 ($\pm$3.8)
Laplacian Eigenmaps	35.8 ($\pm$5.0)	69.2 ($\pm$10.8)

Table 6.4: K-means clustering of face poses applied to PIE-10K for different algorithms. Results are averaged over 10 random K-means initializations.

Methods	Purity (%)	Accuracy (%)
PCA	54.6 ($\pm$1.3)	46.8 ($\pm$1.3)
Nyström Isomap	59.9 ($\pm$1.5)	53.7 ($\pm$4.4)
Col-Sampling Isomap	56.1 ($\pm$1.0)	50.7 ($\pm$3.3)
Laplacian Eigenmaps	39.3 ($\pm$4.9)	74.7 ($\pm$5.1)

Table 6.5: K-means clustering of face poses applied to PIE-35K for different algorithms. Results are averaged over 10 random K-means initializations.

inputs as nearby as possible in the low-dimensional space. When used for clustering, these compact embeddings lead to a few large clusters and several tiny clusters, thus explaining the high accuracy and low purity of the clusters. This indicates poor clustering performance of Laplacian Eigenmaps, since one can achieve even 100% Accuracy simply by grouping all points into a single cluster. However, the Purity of such clustering would be very low. Finally, the improved clustering results of Isomap over PCA for both datasets verify that the manifold of faces is not linear in the input space.

Moreover, we compared the performance of Laplacian Eigenmaps and Isomap embeddings on pose classification.[11] The data was randomly split into a training and a test set, and K-Nearest Neighbor (KNN) was used for classification. $K = 1$ gives lower error than higher K as shown in Table 6.6. Also, the classification error is lower for both exact and approximate Isomap than for Laplacian Eigenmaps, suggesting that neighborhood information is better preserved by Isomap (Tables 6.6 and 6.7). Note that, similar to clustering, the Nyström approximation performs as well as Exact Isomap (Table 6.6). Better clustering and classification results, combined with 2D visualizations, imply that approximate Isomap outperforms exact Laplacian Eigenmaps. Moreover, the Nyström approximation is computationally cheaper and empirically more effective than the Column sampling approximation. Thus, we used Nyström Isomap to generate embeddings for Webfaces-18M.

After learning a face manifold from Webfaces-18M, we analyzed the results with various visualizations. The top row of Figure 6.9 shows the 2D embeddings

[11]KNN only uses nearest neighbor information for classification. Since neighborhoods are considered to be locally linear in the input space, we expect KNN to perform well in the input space. Hence, using KNN to compare low-level embeddings indirectly measures how well nearest neighbor information is preserved.

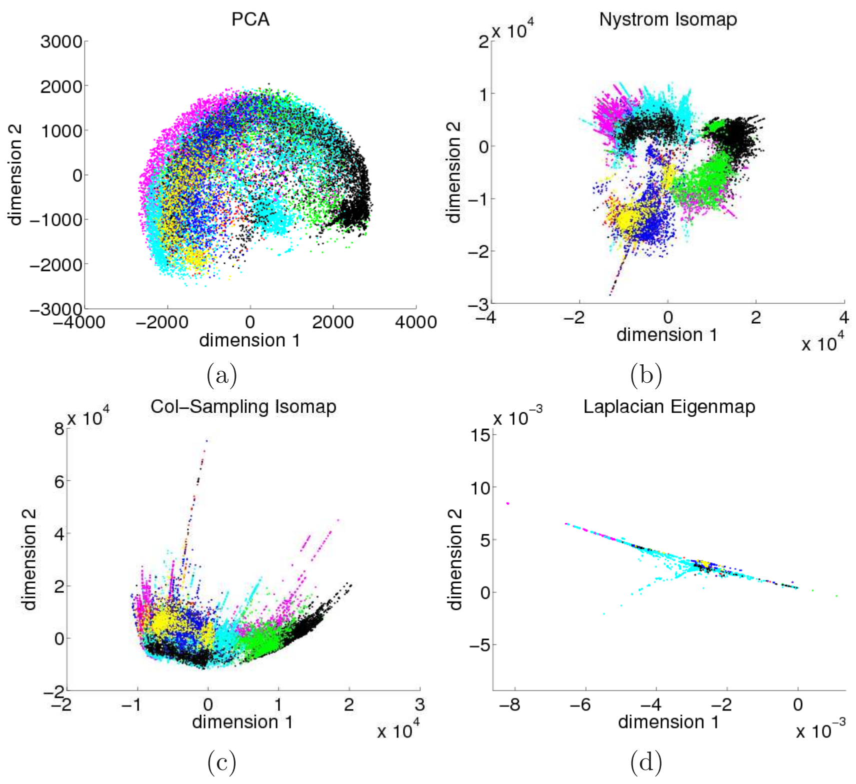

Figure 6.8: Optimal 2D projections of PIE-35K where each point is color coded according to its pose label. (a) PCA projections tend to spread the data to capture maximum variance. (b) Isomap projections with Nyström approximation tend to separate the clusters of different poses while keeping the cluster of each pose compact. (c) Isomap projections with Column sampling approximation have more overlap than with Nyström approximation. (d) Laplacian Eigenmaps projects the data into a very compact range.

Methods	$K = 1$	$K = 3$	$K = 5$
Isomap	10.9 ($\pm$0.5)	14.1 ($\pm$0.7)	15.8 ($\pm$0.3)
Nyström Isomap	11.0 ($\pm$0.5)	14.0 ($\pm$0.6)	15.8 ($\pm$0.6)
Col-Sampling Isomap	12.0 ($\pm$0.4)	15.3 ($\pm$0.6)	16.6 ($\pm$0.5)
Laplacian Eigenmaps	12.7 ($\pm$0.7)	16.6 ($\pm$0.5)	18.9 ($\pm$0.9)

Table 6.6: K-nearest neighbor face pose classification error (%) on PIE-10K subset for different algorithms. Results are averaged over 10 random splits of training and test sets. $K = 1$ gives the lowest error.

Nyström Isomap	Col-Sampling Isomap	Laplacian Eigenmaps
9.8 ($\pm$0.2)	10.3 ($\pm$0.3)	11.1 ($\pm$0.3)

Table 6.7: 1-nearest neighbor face pose classification error (%) on PIE-35K for different algorithms. Results are averaged over 10 random splits of training and test sets.

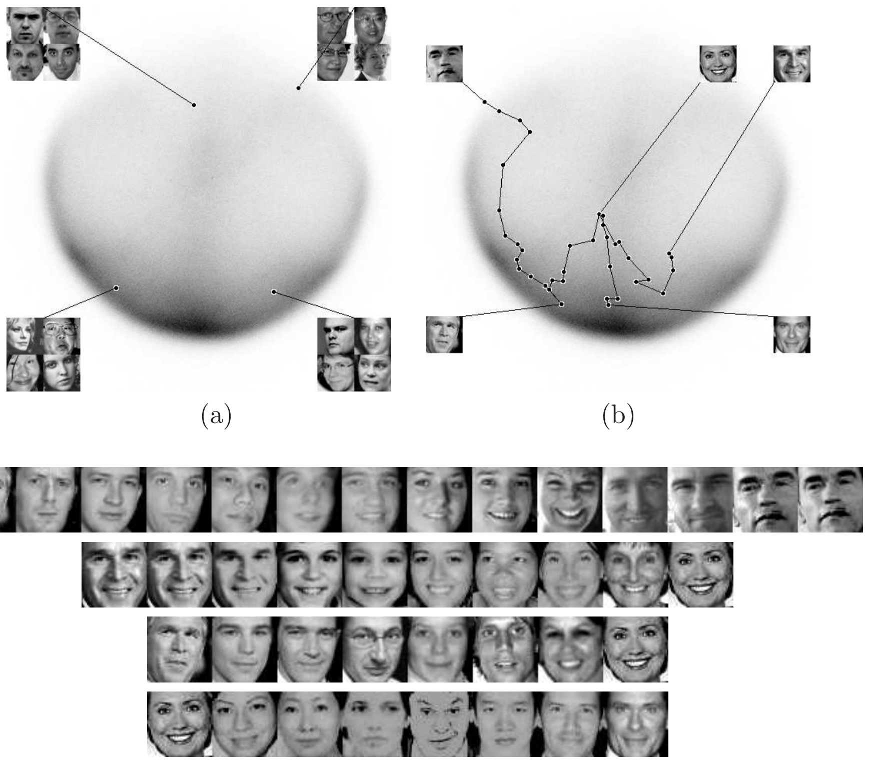

Figure 6.9: 2D embedding of Webfaces-18M using Nyström Isomap (Top row). Darker areas indicate denser manifold regions. (a) Face samples at different locations on the manifold. (b) Approximate geodesic paths between celebrities. (c) Visualization of paths shown in (b).

from Nyström Isomap. The top left figure shows the face samples from various locations in the manifold. It is interesting to see that embeddings tend to cluster the faces by pose. These results support the good clustering performance observed using Isomap on PIE data. Also, two groups (bottom left and top right) with similar poses but different illuminations are projected at different locations. Additionally, since 2D projections are very condensed for 18M points, one can expect more discrimination for higher k, e.g., $k = 100$.

In Figure 6.9, the top right figure shows the shortest paths on the manifold between different public figures. The images along the corresponding paths have smooth transitions as shown in the bottom of the figure. In the limit of infinite samples, Isomap guarantees that the distance along the shortest path between any pair of points will be preserved as Euclidean distance in the embedded space. Even though the paths in the figure are reasonable approximations of straight lines in the embedded space, these results suggest that either (i) 18M faces are perhaps not enough samples to learn the face manifold exactly, or (ii) a low-dimensional manifold of faces may not actually exist (perhaps the data clusters into multiple low dimensional manifolds). It remains an open question as to how we can measure and evaluate these hypotheses, since even very large-scale testing has not provided conclusive evidence.

6.5 Summary

We have presented large-scale nonlinear dimensionality reduction using unsupervised manifold learning. In order to work on a such a large-scale, we first studied sampling based algorithms, presenting an analysis of two techniques for approximating SVD on large dense SPSD matrices and providing a theoretical and empirical comparison. Although the Column sampling method generates more accurate singular values and singular vectors, the Nyström method constructs better low-rank approximations, which are of great practical interest as they do not use the full matrix. Furthermore, our large-scale manifold learning studies reveal that Isomap coupled with the Nyström approximation can effectively extract low-dimensional structure from datasets containing millions of images. Nonetheless, the existence of an underlying manifold of faces remains an open question.

6.6 Bibliography and historical remarks

Manifold learning algorithms are extensions of classical linear dimensionality reduction techniques introduced over a century ago, e.g., Principal Component Analysis (PCA) and Classical Multidimensional Scaling (Pearson, 1901; Cox et al., 2000). Pioneering work on non-linear dimensionality reduction was introduced by Tenenbaum et al. (2000); Roweis and Saul (2000), which led to the development of several related algorithms for manifold learning (Belkin & Niyogi, 2001; Donoho & Grimes, 2003; Weinberger & Saul, 2006). The connection between manifold learning algorithms and Kernel PCA was noted by Ham et al. (2004). The Nyström method was initially introduced as a quadrature method for numerical integration, used to approximate eigenfunction solutions (Nyström, 1928; Baker, 1977). More recently it has been studied for a variety of kernel-based algorithms and other algorithms involving symmetric positive semidefinite matrices (Williams & Seeger, 2000; Fine & Scheinberg, 2002; Fowlkes et al., 2004; Drineas & Mahoney, 2005; Karsmakers et al., 2007; Zhang et al., 2008; Kumar et al., 2009b; Cortes et al., 2010), and in particular for large-scale manifold learning (de Silva & Tenenbaum, 2003; Platt, 2004; Talwalkar et al., 2008). Column sampling techniques have also been analyzed for approximating general rectangular matrices, including notable work by Frieze et al. (1998); Drineas et al. (2006); Deshpande et al. (2006). Initial comparisons between the Nyström method and these more general Column sampling methods were first discussed in Talwalkar et al. (2008); Kumar et al. (2009a).

Bibliography

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., & Mohri, M. (2007). OpenFST: A general and efficient weighted finite-state transducer library. *Conference on Implementation and Application of Automata*.

Baker, C. T. (1977). *The numerical treatment of integral equations*. Oxford: Clarendon Press.

Balasubramanian, M., & Schwartz, E. L. (2002). The Isomap algorithm and topological stability. *Science*, *295*.

Belkin, M., & Niyogi, P. (2001). Laplacian Eigenmaps and spectral techniques for embedding and clustering. *Neural Information Processing Systems*.

Belkin, M., & Niyogi, P. (2006). Convergence of Laplacian Eigenmaps. *Neural Information Processing Systems*.

Chapelle, O., Schölkopf, B., & Zien, A. (Eds.). (2006). *Semi-supervised learning*. Cambridge, MA: MIT Press.

Cortes, C., Mohri, M., & Talwalkar, A. (2010). On the impact of kernel approximation on learning accuracy. *Conference on Artificial Intelligence and Statistics*.

Cox, T. F., Cox, M. A. A., & Cox, T. F. (2000). *Multidimensional scaling*. Chapman & Hall/CRC. 2nd edition.

de Silva, V., & Tenenbaum, J. (2003). Global versus local methods in nonlinear dimensionality reduction. *Neural Information Processing Systems*.

Deshpande, A., Rademacher, L., Vempala, S., & Wang, G. (2006). Matrix approximation and projective clustering via volume sampling. *Symposium on Discrete Algorithms*.

Donoho, D. L., & Grimes, C. (2003). Hessian Eigenmaps: locally linear embedding techniques for high dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, *100*, 5591–5596.

Drineas, P., Kannan, R., & Mahoney, M. W. (2006). Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, *36*.

Drineas, P., & Mahoney, M. W. (2005). On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, *6*, 2153–2175.

Fine, S., & Scheinberg, K. (2002). Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, *2*, 243–264.

Fowlkes, C., Belongie, S., Chung, F., & Malik, J. (2004). Spectral grouping using the Nyström method. *Transactions on Pattern Analysis and Machine Intelligence*, *26*, 214–225.

Frieze, A., Kannan, R., & Vempala, S. (1998). Fast Monte-Carlo algorithms for finding low-rank approximations. *Foundation of Computer Science*.

Golub, G., & Loan, C. V. (1983). *Matrix computations*. Baltimore: Johns Hopkins University Press. 2nd edition.

Gorrell, G. (2006). Generalized Hebbian algorithm for incremental Singular Value Decomposition in natural language processing. *European Chapter of the Association for Computational Linguistics*.

Ham, J., Lee, D. D., Mika, S., & Schölkopf, B. (2004). A kernel view of the dimensionality reduction of manifolds. *International Conference on Machine Learning*.

He, X., Yan, S., Hu, Y., & Niyogi, P. (2005). Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*, 328–340.

Karsmakers, P., Pelckmans, K., Suykens, J., & hamme, J. V. (2007). Fixed-size Kernel Logistic Regression for phoneme classification. *Interspeech*.

Kumar, S., Mohri, M., & Talwalkar, A. (2009a). On sampling-based approximate spectral decomposition. *International Conference on Machine Learning*.

Kumar, S., Mohri, M., & Talwalkar, A. (2009b). Sampling techniques for the Nyström method. *Conference on Artificial Intelligence and Statistics*.

Kumar, S., & Rowley, H. (2010). People Hopper. `http://googleresearch.blogspot.com/2010/03/hopping-on-face-manifold-via-people.html`.

LeCun, Y., & Cortes, C. (1998). The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

Liu, T., Moore, A. W., Gray, A. G., & Yang, K. (2004). An investigation of practical approximate nearest neighbor algorithms. *Neural Information Processing Systems*.

Nyström, E. (1928). Über die praktische auflösung von linearen integralgleichungen mit anwendungen auf randwertaufgaben der potentialtheorie. *Commentationes Physico-Mathematicae*, *4*, 1–52.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, *2*, 559–572.

Platt, J. C. (2004). Fast embedding of sparse similarity graphs. *Neural Information Processing Systems*.

Rokhlin, V., Szlam, A., & Tygert, M. (2009). A randomized algorithm for Principal Component Analysis. *SIAM Journal on Matrix Analysis and Applications*, *31*, 1100–1124.

Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, *290*.

Sim, T., Baker, S., & Bsat, M. (2002). The CMU pose, illumination, and expression database. *Conference on Automatic Face and Gesture Recognition*.

Talwalkar, A. (2010). *Matrix approximation for large-scale learning*. Ph.D. thesis, Computer Science Department, Courant Institute, New York University, New York, NY.

Talwalkar, A., Kumar, S., & Rowley, H. (2008). Large-scale manifold learning. *Conference on Vision and Pattern Recognition*.

Tenenbaum, J., de Silva, V., & Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*.

Weinberger, K. Q., & Saul, L. K. (2006). An introduction to nonlinear dimensionality reduction by maximum variance unfolding. *AAAI Conference on Artificial Intelligence*.

Williams, C. K. I., & Seeger, M. (2000). Using the Nyström method to speed up kernel machines. *Neural Information Processing Systems*.

Zhang, K., Tsang, I., & Kwok, J. (2008). Improved Nyström low-rank approximation and error analysis. *International Conference on Machine Learning*.