A Comparison Study of Graph Laplacian Computation

Michela Marini, Haiyan Cheng, Cristina Garcia-Cardona, Weihong Guo, Sara Hahner, Yuan Liu, Yifei Lou, and Sui Tang

1 Introduction

In recent years, graph signal processing has become popular in many data-driven applications [4, 8, 15, 18, 22, 23], offering a versatile framework for representing and analyzing relationships within complex datasets. By using nodes to signify entities

M. Marini
Department of Mathematics, University of Houston, Houston, TX, USA
e-mail: mmarini2@uh.edu

H. Cheng
School of Computing and Information Sciences, Willamette University, Salem, OR, USA
e-mail: hcheng@willamette.edu

C. Garcia-Cardona
Los Alamos National Laboratory, Los Alamos, NM, USA
e-mail: cgarciac@lanl.gov

W. Guo
Department of Mathematics, Applied Mathematics and Statistics, Case Western Reserve University, Cleveland, OH, USA
e-mail: wxg49@case.edu

S. Hahner
Fraunhofer Institute for Scientific Computing and Algorithms (SCAI), Sankt Augustin, Germany

Y. Liu
Department of Mathematics, Statistics and Physics, Wichita State University, Wichita, KS, USA
e-mail: yuan.liu@wichita.edu

Y. Lou (✉)
Department of Mathematics & School of Data Science and Society, University of North Carolina at Chapel Hill,, Chapel Hill, NC, USA
e-mail: yflou@unc.edu

S. Tang
Department of Mathematics, University of California Santa Barbara, Santa Barbara, CA, USA
e-mail: suitang@ucsb.edu

C. Garcia-Cardona, H. Lee (eds.), *Advances in Data Science*, Association for Women in Mathematics Series 37, https://doi.org/10.1007/978-3-031-87804-6_8

and edges to denote connections between them, graphs can model a wide array of structures, from social networks and biological systems to transportation grids and recommendation engines.

Consider a collection of data points $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathbb{R}^v$, where n is the number of points and v is the dimension of each feature vector. One constructs a graph $G(V, E)$ by treating each point as a vertex $v_i \in V$, $i = 1, \ldots, n$, and E an edge connectivity representing specific relations between vertices. E can be represented by a matrix, called an adjacency matrix. Specifically, for a graph with n nodes, the adjacency matrix, denoted by A, is an $n \times n$ matrix where each element a_{ij} indicates whether there is an edge from node i to node j. The value of a_{ij} is typically either 1 (indicating the connection) or 0 (no connection). A generalization of the adjacency matrix is a similarity matrix W associated with a weighted graph where each edge is characterized by a real weight w_{ij} representing application-specific meanings, usually a measure of how similar nodes i and j are. This paper focuses on an undirected and unsigned graph corresponding to a symmetric and nonnegative weight function, i.e., $w_{ij} = w_{ji} \geq 0, \forall 1 \leq i, j \leq n$.

The graph Laplacian, derived from the similarity matrix of a weighted graph, is a fundamental tool in spectral graph theory [11]. Let the degree matrix D be a diagonal matrix where each diagonal element is defined by $d_{ii} = \sum_j w_{ij}$. The unnormalized graph Laplacian L, defined as $L = D - W$, encapsulates important structural properties of the graph, such as connectivity and the presence of clusters. For data science applications, it is widely recognized [4, 18] the computational and performance advantages of deploying the symmetric normalized Laplacian, which is defined as

$$L_s = I - D^{-1/2} W D^{-1/2}. \tag{1}$$

The eigenvalues and eigenvectors of L_s are particularly useful, providing insights into graph partitioning [9], clustering [4, 19, 22, 26], machine learning [8, 13], and the behavior of diffusion processes on the graph [10].

However, it is computationally intensive to obtain the similarity matrix and the graph Laplacian, often becoming a bottleneck in dealing with "big data." Specifically, the computational complexity of constructing a graph Laplacian is of the order $O(n^2)$, making it intractable when n is extremely large. In addition, when the graph Laplacian is used in certain applications [4, 23], the eigendecomposition and/or singular value decomposition (SVD) is often required, which is in the computational complexity of $O(n^3)$. Consequently, accelerating the construction of the graph Laplacian together with its decompositions is essential for handling large-scale graph-based applications.

This paper studies three methods to approximate L_s. The first method, called K-nearest neighbors (KNN), involves creating a sparse approximation by computing a small number of pairwise weight functions for each node, resulting in a sparse matrix. The other two methods focus on low-rank approximations and are called Nyström method [14] and its variant using the QR decomposition [6]. Our empirical evaluation of the methods yields the following observations:

- Nyström methods (the original one and its QR-based variant) provide good approximations to the eigendecomposition of the Laplacian for the fully connected graph while considerably reducing computation times since they require computations for only a handful of samples in the dataset. This is observed in both benchmarks and high-dimensional datasets.
- Both Nyström-based methods are particularly advantageous when an eigendecomposition is required for downstream tasks, as they provide efficient algorithms for computing accurate approximations without increasing time demands.
- The KNN method provides an excellent approximation to the Laplacian of the fully connected graph (given that the similarity metric is sufficiently smooth), but requires computations over the entire dataset, which can become intractable for datasets with a large number of nodes.

The rest of the paper is organized as follows. Section 2 provides a brief review of the methods: KNN, Nyström, and QR-based Nyström. We then investigate the performance of these approximations in Sect. 3 in terms of accuracy to approximate the fully connected graph, computational time, and efficiency in applications of classification, clustering, and CT reconstruction. Finally, the conclusions are given in Sect. 4.

2 Method Review

A fully connected weighted graph can be represented via a dense weight matrix W of dimensions $n \times n$, where every pair of nodes is connected with an assigned similarity value. In this work, we use the Gaussian similarity metric, where each weight entry is defined as

$$w_{ij} = \exp\left\{\frac{-d(\mathbf{x}_i, \mathbf{x}_j)^2}{2\sigma^2}\right\}, \quad i, j = 1, \ldots, n, \tag{2}$$

with $d(\mathbf{x}_i, \mathbf{x}_j)$ being the Euclidean distance between the two samples (i.e., vertices) $\mathbf{x}_i$ and $\mathbf{x}_j$, which can be computed as $d_E(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$, i.e., the conventional measure for calculating the distance between two points in the Euclidean space. Note that $\sigma > 0$ controls the smoothness of the similarity metric, providing more drastic differences when its value is small and more gradual transitions when its value is large. Note that the diagonal element $w_{ii} = 1$ follows the definition in Eq. (2), which is reasonable due to self-similarity.

When dealing with big data, e.g., hyperspectral data where the number of pixels in the image could be in the order of 10^6, the weight matrix presents computational challenges and requires significant storage space. We review three ways to approximate the weight matrix, namely, K-nearest neighbor [12], Nyström method [14], and QR-based Nyström decomposition [6]. In the experimental section, we compare their performance in terms of accuracy and efficiency.

2.1 K-Nearest Neighbor Graph

The K-nearest neighbor (KNN) graph is frequently used in machine learning and data analysis, particularly in pattern recognition, classification, and clustering tasks [24, 27, 29]. As the name suggests, KNN constructs a graph by connecting each node to its K-nearest neighbors based on a chosen distance metric. To do this, one must first determine an appropriate distance metric and select a value for K.

For each data point, a distance metric is computed between this point and the other points, followed by Eq. (2) to obtain the similarity measures between any pair. Subsequently, weights are only stored for the K-nearest neighbors, corresponding to the K largest similarity values. This process results in a sparse weight matrix W with each row having at most $K (\ll n)$ nonzero elements.

The naive KNN does not guarantee a symmetric matrix, since the node i being in the top K neighbor of j does not entail j being in the top K neighbor of i. To make the weight symmetric, we adopt a simple approach by taking the average of the weight and its transpose, i.e., $W \leftarrow \frac{1}{2}(W + W^\top)$. Another alternative is the mutual KNN [20], which is out of the scope of this paper.

2.2 Nyström Method

To reduce the time/space complexity, Fowlkes et al. [14] proposed the Nyström method to approximate the eigenvalues and eigenvectors of $W \in \mathbb{R}^{n\times n}$ by using only p sampled data points with $p \ll n$. Up to permutations, we adopt a block-matrix form to represent the weight matrix W as follows:

$$W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}, \tag{3}$$

where $W_{11} \in \mathbb{R}^{p\times p}$ is the weight (similarity) matrix between the sampled data points, $W_{12} = W_{21}^\top$ is the one between the sampled points and the unsampled points, and W_{22} is the one between the unsampled points. The idea of Nyström extension is to approximate the matrix W and its corresponding normalized graph Laplacian, L_s defined in Eq. (1), using W_{11} and W_{12}, thereby avoiding the computation of the relatively large matrix W_{22}. Since the matrix L_s involves the degree matrix, we begin by normalizing W so that its degree matrix becomes the identity. In particular, we define a matrix

$$\overline{W} = \begin{bmatrix} W_{11} & W_{21}^\top \\ W_{21} & W_{21} W_{11}^{-1} W_{21}^\top \end{bmatrix}, \tag{4}$$

and its row-sum vector in a block form:

$$\begin{bmatrix}\mathbf{d}_1\\ \mathbf{d}_2\end{bmatrix} = \overline{W}\mathbf{1}_n = \begin{bmatrix}W_{11} & W_{21}^\top\\ W_{21} & W_{21}W_{11}^{-1}W_{21}^\top\end{bmatrix}\begin{bmatrix}\mathbf{1}_p\\ \mathbf{1}_{n-p}\end{bmatrix},$$

where $\mathbf{1}_k$ denotes the k-dimensional all one vector. Denoting $\mathbf{s}_1 = \sqrt{\mathbf{d}_1}$ and $\mathbf{s}_2 = \sqrt{\mathbf{d}_2}$, we can normalize the matrices W_{11} and W_{21} by

$$\widetilde{W}_{11} = W_{11} \oslash (\mathbf{s}_1\mathbf{s}_1^\top) \quad \widetilde{W}_{21} = W_{21} \oslash (\mathbf{s}_1\mathbf{s}_2^\top), \tag{5}$$

where $\oslash$ denotes the componentwise division. In the same block format as $\overline{W}$, we define

$$\widetilde{W} = \begin{bmatrix}\widetilde{W}_{11} & \widetilde{W}_{21}^\top\\ \widetilde{W}_{21} & \widetilde{W}_{21}\widetilde{W}_{11}^{-1}\widetilde{W}_{21}^\top\end{bmatrix}. \tag{6}$$

By definition, the degree matrix corresponding to $\widetilde{W}$ becomes the identity, and the symmetric normalized graph Laplacian becomes

$$\widetilde{L}_s = I - \widetilde{W}. \tag{7}$$

Next, we describe the SVD of $\overline{W}$ and use it to represent the symmetric normalized graph Laplacian $\widetilde{L}_s$. We assume $\widetilde{W}_{11}$ is positive definite (by choosing a proper value of σ in Eq. (2)); then it is invertible and we further denote $\widetilde{W}_{11}^{1/2}$ as its square root. We can express $\widetilde{W}$ in the following way:

$$\begin{aligned}\widetilde{W} &= \begin{bmatrix}\widetilde{W}_{11}\\ \widetilde{W}_{21}\end{bmatrix}\widetilde{W}_{11}^{-1}[\widetilde{W}_{11} \quad \widetilde{W}_{21}^\top]\\ &= \left\{\begin{bmatrix}\widetilde{W}_{11}\\ \widetilde{W}_{21}\end{bmatrix}\widetilde{W}_{11}^{-1/2}U\Sigma^{-1/2}\right\}\Sigma\left\{\Sigma^{-1/2}U^\top\widetilde{W}_{11}^{-1/2}[\widetilde{W}_{11} \quad \widetilde{W}_{21}^\top]\right\},\end{aligned} \tag{8}$$

for any diagonal matrix Σ and unitary matrix U, both of which can be determined by the requirement that $V^\top V = I$ with

$$V := \begin{bmatrix}\widetilde{W}_{11}\\ \widetilde{W}_{21}\end{bmatrix}\widetilde{W}_{11}^{-1/2}U\Sigma^{-1/2}.$$

We elaborate on this requirement by expressing it into

$$I = V^\top V = \left\{\Sigma^{-1/2}U^\top\widetilde{W}_{11}^{-1/2}[\widetilde{W}_{11} \quad \widetilde{W}_{21}^\top]\right\}\left\{\begin{bmatrix}\widetilde{W}_{11}\\ \widetilde{W}_{21}\end{bmatrix}\widetilde{W}_{11}^{-1/2}U\Sigma^{-1/2}\right\}.$$

Multiplying the above equation from the left by $U\Sigma^{1/2}$ and from the right by $\Sigma^{1/2}U^\top$ yields

$$U\Sigma U^\top = \widetilde{W}_{11} + \widetilde{W}_{11}^{-1/2}\,\widetilde{W}_{21}^\top\,\widetilde{W}_{21}\,\widetilde{W}_{11}^{-1/2},$$

which implies that U and Σ can be obtained by the SVD of the matrix $\widetilde{W}_{11} + \widetilde{W}_{11}^{-1/2}\widetilde{W}_{21}^\top\widetilde{W}_{21}\widetilde{W}_{11}^{-1/2}$. In summary, we have $\widetilde{W} = V\Sigma V^\top$ with $V^\top V = I$.

Using the SVD of $\widetilde{W}$, we further approximate $\widetilde{L}_s$, defined in Eq. (7), by

$$\widetilde{L}_s \approx V(I - \Sigma)V^\top = V\Lambda V^\top, \tag{9}$$

with $\Lambda = I - \Sigma$. This is an approximation, as $VV^\top$ is generally not the identity matrix. An improvement, originally suggested in [7], is to use the decomposition to approximate $I - L_s$ instead, i.e., $\widetilde{L}_s \approx I - V\Sigma V^\top$. We denote this alternative approximation as Nyström $(I - L_s)$. In Sect. 3, we compare the performance of the two Nyström-based alternatives to compute $\widetilde{L}_s$ through numerical experiments.

Overall, the Nyström approach significantly reduces the computational costs by computing pairwise similarities only for a subset of the dataset, resulting in the computational complexity and storage requirements of $O(n)$ instead of $O(n^2)$, as p is negligible compared to n.

2.3 QR-Based Nyström Decomposition

The Nyström method requires $\widetilde{W}_{11}$ to be positive definite so that its square root is well-defined in Eq. (8) to calculate the SVD of the corresponding normalized graph Laplacian. If $\widetilde{W}_{11}$ is indefinite, Fowles et al. [14] provided a feasible solution based on [3], but unfortunately, this approach incurs additional computational cost and is prone to numerical errors.

Inspired by the work of [1] that used a recompression technique in [2] for computing a fully connected graph Laplacian, Budd et al. [6] employed the QR decomposition instead of SVD when approximating the normalized graph Laplacian. Specifically, we consider the thin QR decomposition of

$$\begin{bmatrix}\widetilde{W}_{11}\\ \widetilde{W}_{21}\end{bmatrix} = QR, \tag{10}$$

where $\widetilde{W}_{11}$ and $\widetilde{W}_{12}$ are obtained in Eq. (5), $Q \in \mathbb{R}^{n\times p}$ is orthonormal, and $R \in \mathbb{R}^{p\times p}$ is upper triangular. Then, we have the eigendecomposition:

$$R\widetilde{W}_{11}^{-1}R^\top = \Phi\Sigma\Phi^\top, \tag{11}$$

where $\Phi \in \mathbb{R}^{p\times p}$ is orthonormal and $\Sigma \in \mathbb{R}^{p\times p}$ is diagonal. We define $\Psi = Q\Phi$, which is orthonormal and adopt the following eigendecomposition of the symmetric normalized Laplacian:

Table 1 The computational complexity for KNN and Nyström methods for obtaining a normalized graph Laplacian of size $n \times n$, with K as the internal parameter for KNN and p for both Nyström methods

Method	Complexity
KNN	$O(Kn)$
Nyström	$O(np^2 + p^3)$
QR	$O(n^2 p + p^3)$

$$L_s \approx \Psi(I - \Sigma)\Psi^\top = \Psi \Lambda \Psi^\top. \tag{12}$$

Please refer to [2, 6] for more details.

Similar to the Nyström case, the decomposition can be used to approximate $I - L_s$ instead, i.e., $L_s \approx I - \Psi \Sigma \Psi^\top$. We denote this alternative approximation as QR $(I - L_s)$. In Sect. 3, we compare the performance of the two QR-based alternatives to compute $\widetilde{L}_s$ through numerical experiments.

2.4 Summary

The choice of method depends on the specific requirements of the task, such as the size of the dataset, the desired accuracy, and the available computational resources. KNN is a simple and intuitive method for computing the weight matrix. It is effective for processing data with a clear local structure, but it can be sensitive to the choice of K and less effective for large, nonuniform datasets. Both Nyström methods can achieve good approximations for the symmetric normalized graph Laplacian with a relatively small number of columns, though random selection can sometimes lead to poor performance. The QR variant of the Nyström method enhances numerical robustness in the approximation but comes with higher computational costs compared to the standard Nyström method. The computational complexity of each method is provided in Table 1.

3 Numerical Experiments

We conduct numerical experiments on two benchmark datasets and one high-dimensional dataset to evaluate the efficacy of three graph Laplacian computation approaches, including KNN, Nyström, and QR-based Nyström (QR in short). The two benchmark datasets are obtained from the Scikit-learn library [21], while a high-dimensional dataset is the low-dose CT dataset [17] as processed for CT reconstruction in [28].

3.1 Benchmark Datasets

We use two benchmark datasets from Scikit-learn, namely, the two-moon and digits datasets. For each dataset, we compute (i) a fully connected graph with Gaussian similarity for the weight matrix W defined in Eq. (2), (ii) the symmetric normalized Laplacian L_s defined in Eq. (1), and (iii) the corresponding eigendecomposition via the `eigh` function of the `linalg` utilities of the NumPy Python package. This matrix L_s and its eigendecomposition become the *ground truth* with respect to which the performance of the methods is evaluated. We compare the performance of the three aforementioned methods, including the variant of approximating $(I - L_s)$. Note that KNN computes a sparse approximation to the weight matrix W, followed by the symmetric normalization to obtain the graph Laplacian L_s. In this case, we again compute the corresponding eigendecomposition via the `eigh` function of the `linalg` utilities of NumPy. In contrast, Nyström and QR-based Nyström directly compute an eigendecomposition of L_s.

The results reported include a comparison of the eigendecomposition obtained for each method, approximation errors, computation times, and accuracy obtained for unsupervised (clustering) and supervised (classification) tasks using the eigendecomposition as a pre-processing step. For the eigendecomposition, we report results obtained under different σ^2 values in Eq. (2) to reveal a stability issue in the original Nyström method. For the remaining comparisons, we examine two values of σ^2, and for each value, we vary the number of neighbors (K) in KNN and the number of sample data points (p) in Nyström methods. For each combination of parameters, we report mean and standard deviations over 30 repetitions of the whole processing pipeline, consisting of the following steps:

1. Generate data.
2. Split into training (70%) and testing (30%) partitions.
3. Construct Laplacians and their eigendecompositions using the training partition.
4. Evaluate clustering accuracy (over training partition).
5. Evaluate classification accuracy (over testing partition).

Approximation Error The approximation error is computed in terms of the relative Frobenius distance:

$$E_r = \frac{\|\widehat{L}_s - L_s\|_F}{\|L_s\|_F} , \tag{13}$$

where L_s is the ground truth, i.e., symmetric normalized Laplacian for the fully connected graph, and $\widehat{L}_s$ is the approximation, which, as a reminder, corresponds to

- KNN: $\widehat{L}_s = I - \widetilde{D}^{1/2}\widetilde{W}\widetilde{D}^{1/2}$, with $\widetilde{W}$ the similarity matrix including only K-nearest neighbors.
- Nyström: $\widehat{L}_s = V \Lambda V^\top$, computed using p sampled data points.
- Nyström $(I - L_s)$: $\widehat{L}_s = I - V \Sigma V^\top$, computed using p sampled data points.

- QR-based Nyström: $\widehat{L}_s = \Psi \Lambda \Psi^\top$, computed using p sampled data points.
- QR-based Nyström $(I - L_s)$: $\widehat{L}_s = I - \Psi \Sigma \Psi^\top$, computed using p sampled data points.

Computation Time Computation times reported were obtained on a 2.4 GHz 8-Core Intel Core i9 MacBook Pro.

Clustering Accuracy We use spectral clustering [26], i.e., K-means over the eigenvectors of L_s, as an unsupervised graph-based method to partition data into clusters. In each case, we only select a handful (5–25) of the top eigenvectors (i.e., the eigenvectors associated with the 5–25 smallest eigenvalues of matrix L_s). Since this is an unsupervised method, we do not make use of the class labels. To evaluate the accuracy, we only use the training data (i.e., the data used to build the graph Laplacian) and make use of the Scikit-learn [21] `rand_score` metric, which computes the *rand index*, a similarity measure between two clusterings based on "considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings."

Classification Accuracy We apply the support vector machine (SVM) technique for classification [16] using the Scikit-learn [21] functionality. SVM classification is a supervised learning algorithm that tries to find a maximum margin separating hyperplane, i.e., a hyperplane that separates the classes and has the maximum distance between data points in disparate classes. Instead of using the data points in the original domain, we project them onto a subspace defined by the eigenvectors of a Laplacian matrix, i.e., $\widetilde{X} = XU^\top$, where X is a matrix with rows corresponding to the data points and U is the matrix that is composed of eigenvectors of L_s. We only use a subset of eigenvectors corresponding to the dimensionality of the data. In this way, we can project both training and testing partitions. We also use a linear kernel, to evaluate the usefulness of the eigendecomposition as a pre-processing mechanism. To evaluate the accuracy, given that we know the true labels, we use the testing data and make use of the Scikit-learn `accuracy_score` metric, which computes the fraction of correctly classified samples.

3.1.1 Two-Moons Dataset

The *two-moons* dataset comprises a total of 2000 samples. Each sample is a point in a 2D plane, following the arch of a moon. As shown in Fig. 1, the dataset is divided into two classes, purple and yellow points, each containing 1000 samples. Additionally, each class comprises 500 points where the true moon samples have been perturbed with a 10% noise level, and another 500 points where the true moon samples have been perturbed with a 20% noise level.

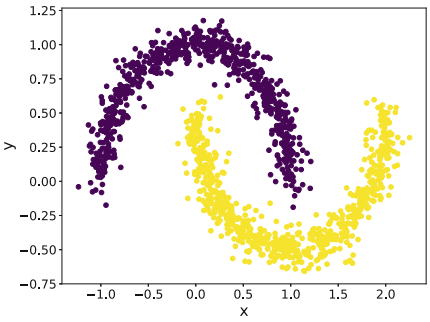

Fig. 1 Two-moons dataset from one random realization of the noise distribution. Each sample is a 2D vector belonging to one of two classes: either purple or yellow

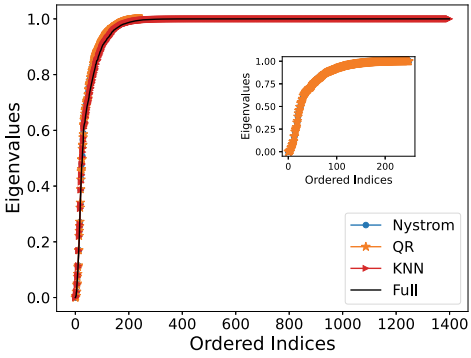

Fig. 2 Eigenvalues of the symmetric normalized Laplacian obtained by KNN ($K = 10$) and Nyström methods ($p = 250$) on the two-moons dataset with $\sigma^2 = 0.01$. Note that Nyström methods completely overlap and only QR, which lays on top of original Nyström, is visible in the plots

Eigendecomposition

Figure 2 compares the eigenvalues obtained by the three methods with $\sigma^2 = 0.01$, $K = 10$ nearest neighbors for KNN and $p = 250$ sampled points for both Nyström methods. It is clear that all the eigenvalues approximate the ones for the fully connected graph (labeled by "Full" in Fig. 2). The inset is included to remark that Nyström methods produce a rank p approximation to the eigendecomposition, thereby making only p eigenvalues available for these methods. Similarly, Fig. 3 compares the Nyström and Nyström ($I - L_s$) approximations (left) as well as QR and QR ($I - L_s$) approximations (right). Both methods with two approximation variants display a good agreement with the eigenvalues of the fully connected graph.

We then examine the top three eigenvectors (i.e., the eigenvectors associated with the smallest eigenvalues) of L_s obtained by all the methods in Fig. 4. As the original two-moons data is in 2D, we can plot the distribution of the training set in the x-y plane and color each point according to the value of a specific eigenvector. The row ordering of the input data X establishes the row correspondence to the eigenvector components. Note that the first eigenvector (first row), in which L_s is related to the normalized degree [26], remains consistent between fully connected graph and Nyström approximations. In contrast, it remains almost constant for KNN, as expected, since the normalized degree should be similar for graphs with the same number of nearest neighbors. Likewise, Fig. 5 compares the Nyström , Nyström

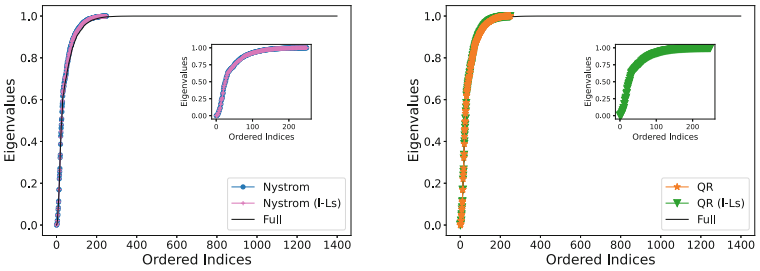

Fig. 3 Eigenvalues of the symmetric normalized Laplacian obtained by Nyström methods ($p = 250$) on the two-moons dataset with $\sigma^2 = 0.01$. Note that the methods completely overlap and only $(I - L_s)$ variants, which lay on top of the direct L_s approximation, are visible in the plots

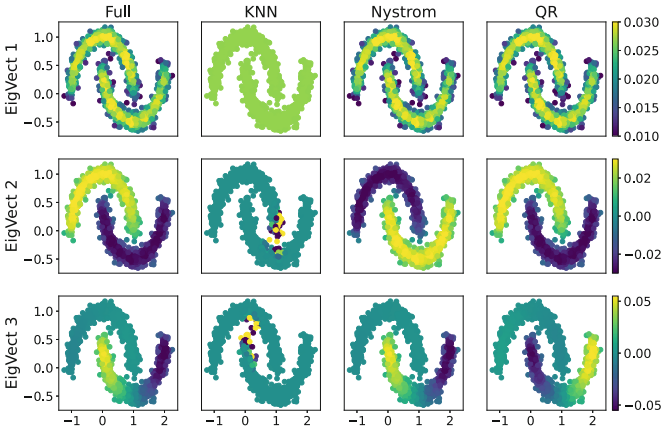

Fig. 4 Top three eigenvectors of the symmetric normalized Laplacian obtained by KNN ($K = 10$) and Nyström methods ($p = 250$) on the two-moons dataset with $\sigma^2 = 0.01$

$(I - L_s)$, QR, and QR $(I - L_s)$ approximations, showing a good agreement among these methods. In summary, Figs. 4 and 5 illustrate that, aside from sign differences in the eigenvectors, all the Nyström variants produce a good approximation to the first eigenvectors. The KNN method, on the other hand, produces much more localized patterns. The errors in the approximations given by Eq. (13) are 0.127935 for KNN, 0.927003 for Nyström , 0.022307 for Nyström $(I - L_s)$, 0.926823 for QR, and 0.021647 for QR $(I - L_s)$. From these error estimations, it is clear that the $(I - L_s)$ variant of the Nyström methods produces much better approximations to the full symmetric normalized Laplacian than the direct L_s approximations.

We investigate the eigenvalues obtained by the competing methods under different values of σ^2; specifically, $\sigma^2 = 0.005, 0.01, 0.07$ and 0.1 are considered in Fig. 6, showing that the smaller σ^2 is, the larger error to the fully connected graph is made by the Nyström approximations. For simplicity, we omit the $(I - L_s)$ approximation variants from Fig. 6, because they fall on top of the graphs for the

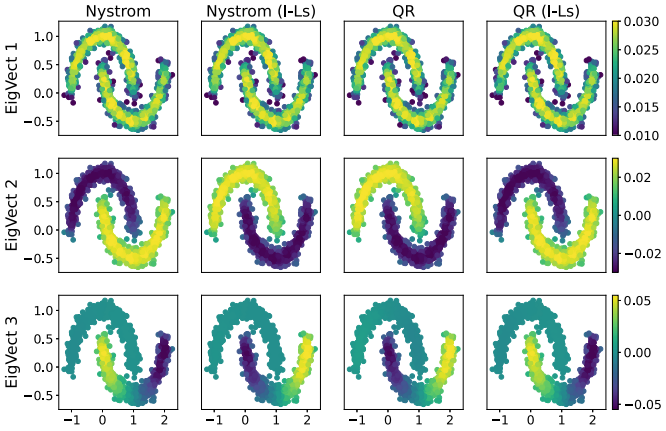

Fig. 5 Top three eigenvectors of the symmetric normalized Laplacian obtained by Nyström methods ($p = 250$) with the two approximation variants on the two-moons dataset with $\sigma^2 = 0.01$

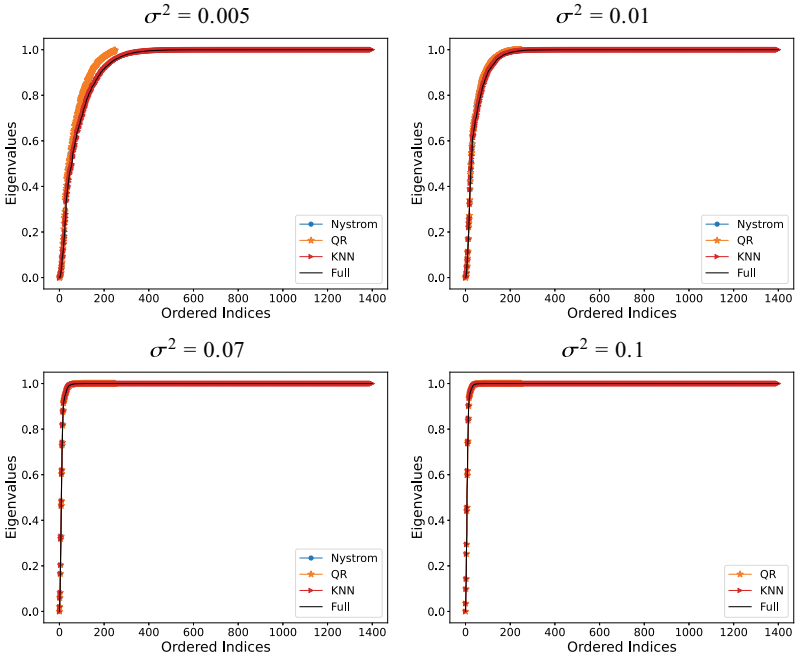

Fig. 6 Eigenvalues of the symmetric normalized Laplacian obtained by KNN ($K = 10$) and Nyström methods ($p = 250$) on the two-moons dataset under different values of σ^2. Note that Nyström methods completely overlap and only QR, which lays on top of original Nyström , is visible in the plots

Table 2 Comparison of the approximations errors, i.e., E_r defined in (13), made by KNN ($K = 10$) and Nyström method ($p = 250$) on the two-moons dataset under different values of σ^2. NaN indicates that the original Nyström method fails at $\sigma^2 = 0.1$ when the submatrix W_{11} is not positive definite

	σ^2			
Method	0.005	0.01	0.07	0.1
KNN	0.180776	0.127935	0.066262	0.058065
Nyström	0.940897	0.927003	0.911308	NaN
Nyström ($I - L_s$)	0.068371	0.022307	0.000006	NaN
QR	0.941037	0.926823	0.911150	0.910177
QR ($I - L_s$)	0.069773	0.021647	0.000008	0.000005

direct L_s approximation when plotted. On the other hand, both variants of the original Nyström method fail for larger values of σ^2, e.g., $\sigma^2 = 0.1$ and $p = 250$ used here, as the submatrix W_{11} is not positive definite. Note that both QR-based variants succeed in this case. Table 2 records the approximation errors for these four values of σ^2. Note that, in general, the $(I - L_s)$ variants yield better approximation results.

Approximation Errors

The approximation errors with respect to a range of K-nearest neighbors in KNN and p sampled data points in both Nyström methods, using both approximation variants, are plotted in Fig. 7 for $\sigma^2 = 0.01$ and $\sigma^2 = 0.07$. The results are averaged over 30 random trials. Since the ranges of K and p are different, the plots include two x-axis: the top one in red corresponds to the K values for KNN, while the bottom one in black corresponds to the p values for Nyström methods. Figure 7 clearly illustrates that the approximation given by the Nyström methods improves as the number of sample points p increases. It also shows that the Nyström method does not converge for larger values of p, where only results for $p \leq 250$ can be computed. Since the original Nyström and QR mostly overlap, it is difficult to observe the lack of convergence of the original Nyström from these error plots. However, the other plots, especially Fig. 9, make this observation more apparent. The approximation errors for the KNN method are generally smaller than the Nyström methods (for the direct L_s approximation) and are relatively independent of K. The Nyström methods that approximate $(I - L_s)$ produce smaller errors, compared to KNN. The performance of Nyström methods on downstream tasks involving the eigendecomposition is better than the KNN method as shown in Figs. 9 and 10.

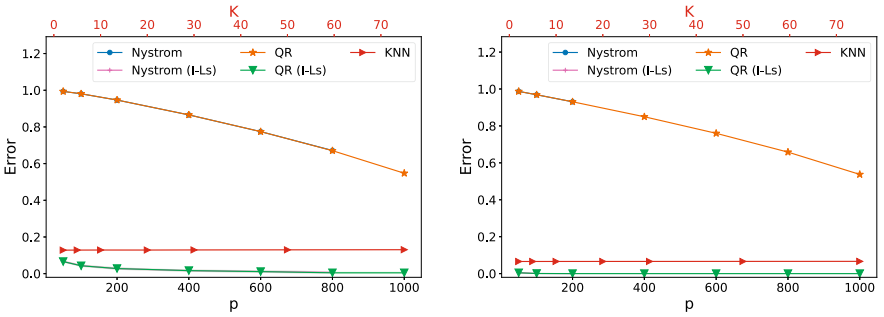

Fig. 7 Error in L_s approximation for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 0.01$ (left) and $\sigma^2 = 0.07$ (right), on the two-moons dataset. The results are averaged over 30 random trials and computed means are reported. The standard deviation computed is very small, with practically no-shaded region distinguishable. Note that Nyström methods completely overlap (up to where the original Nyström is stable, i.e., $p \leq 800$ (left) and $p \leq 200$ (right)), and only QR results, which fall on top of the original Nyström (same phenomenon for the QR methods), are visible in the plots

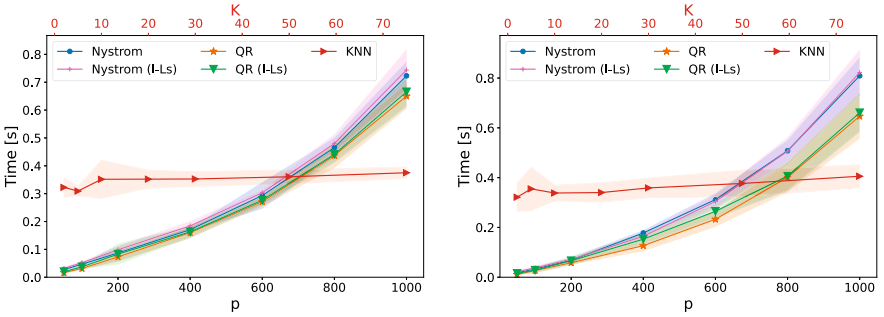

Fig. 8 Computation times for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 0.01$ (left) and $\sigma^2 = 0.07$ (right), on the two-moons dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials

Computation Time

Under the same setup as the approximation error, the computation times are plotted in Fig. 8, where the standard deviations calculated over 30 random trials are depicted as a shaded region. Note that the times reported for KNN include the eigendecomposition stage, which is naturally included in the Nyström class. Figure 8 shows that the QR-based Nyström is slightly faster than the original Nyström method, and their difference becomes larger as p or σ^2 increases. In addition, the KNN method, utilizing the nearest neighbors routine from the `giotto-tda` Python package [25], ensures stable computation times, remaining almost constant across the range of $K \in [2, 75]$ most probably due to its exploitation of multi-core parallelism. Figure 8

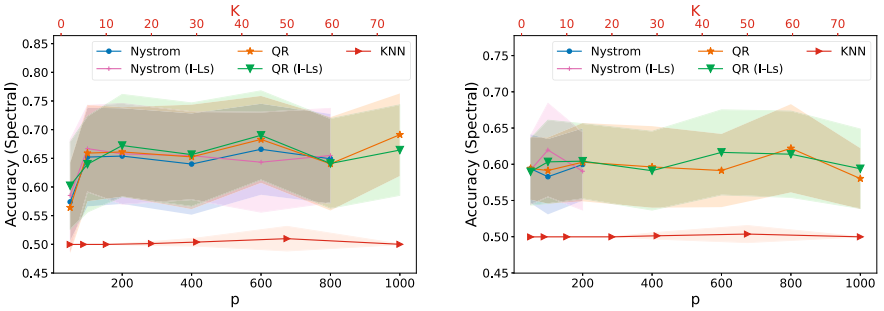

Fig. 9 Accuracy of spectral clustering for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 0.01$ (left) and $\sigma^2 = 0.07$ (right), on the two-moons dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials

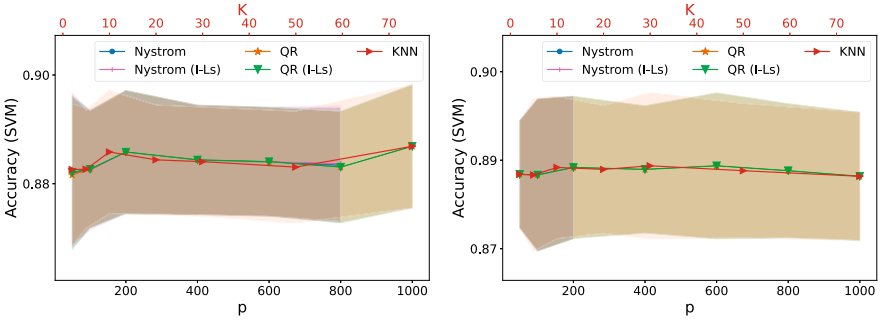

Fig. 10 Accuracy of SVM classification for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 0.01$ (left) and $\sigma^2 = 0.07$ (right), for the two-moons dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials. Note that Nyström methods completely overlap (up to where the original Nyström is stable, i.e., $p \leq 800$ (left) and $p \leq 200$ (right)), and only QR, which lays on top of original Nyström, is visible in the plots

reveals that there is a range where substantial computation savings can be obtained by using the Nyström approximation methods, without a significant sacrifice in performance (see accuracy plots, e.g., Figs. 9 and 10).

Unsupervised Task

We report the performance of the weight approximation methods in a downstream task: unsupervised clustering. Specifically, averaged accuracy results obtained by spectral clustering over 30 random trials are plotted in Fig. 9 for $\sigma^2 = 0.01$ and $\sigma^2 = 0.07$. The standard deviations calculated over the random trials are depicted

Fig. 11 Representative samples from each of the ten-class digits dataset. Each sample is an 8×8 pattern that can be flattened to a 64-dimensional vector. The training set used has about 1250 samples

as a shaded region in the plots. Given that the eigenvectors tend to be more localized in KNN, 25 eigenvectors are used for the spectral clustering, while only five eigenvectors are used for Nyström methods. It is clear in Fig. 9 that projecting on the eigendecomposition of the Nyström methods produces better results than KNN, but no major improvements are observed for approximations using larger K or p. These plots also make more evident that no results are reported for Nyström $p > 800$ (left plot) and for $p > 250$ (right plot) due to the invalid partial computations (i.e., submatrix W_{11} not positive definite or unstable inversion).

Supervised Task

Another downstream task given by the SVM classification is examined in Fig. 10, showing that supervised learning contributes to a large improvement in the classification results compared to unsupervised clustering. It is also interesting to note that although the Nyström methods that directly approximate L_s yield larger approximation errors than KNN (see Fig. 7), the classification accuracy is similar and relatively high for all the weight approximation methods, probably due to the supervised nature of this task.

3.1.2 Digits Dataset

The *digits* dataset comprises a total of 1797 images of handwritten digits ranging from 0 to 9. Each image is of dimension 8×8 and hence can be represented by a 64-dimensional array of gray-scale intensity values, vectorized from a 2D image. This dataset is a copy of the test set of the UCI ML handwritten digits datasets.[1] An illustration of the images in each class can be found in Fig. 11.

[1] https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

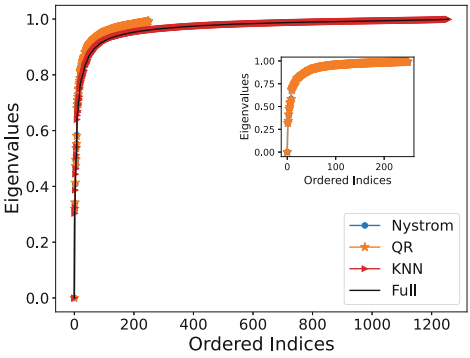

Fig. 12 Eigenvalues of the symmetric normalized Laplacian for the digits dataset for each of the methods with $\sigma^2 = 1.0$, $K = 10$ for KNN, and $p = 250$ for Nyström methods. Note that Nyström methods completely overlap and only QR, which lays on top of original Nyström, is visible in the plots

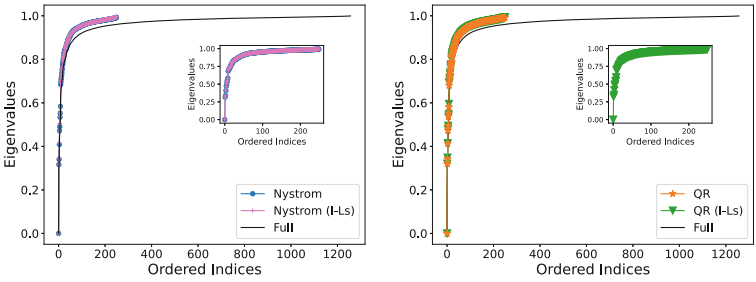

Fig. 13 Eigenvalues of the symmetric normalized Laplacian obtained by Nyström methods ($p = 250$) on the digits dataset with $\sigma^2 = 1.0$. Note that the methods completely overlap and only $(I - L_s)$ variants, which lay on top of the direct L_s approximation, are visible in the plots

Eigendecomposition

Figure 12 compares the eigenvalues obtained by three methods with $\sigma^2 = 1.0$, $K = 10$ nearest neighbors for KNN, and $p = 250$ sampled points for both Nyström methods. All the eigenvalues approximate the ones for the fully connected graph, except that the Nyström methods start to show a slight deviation from the ground truth. Figure 13 compares the Nyström, Nyström $(I - L_s)$, QR, and QR $(I - L_s)$ approximations, showing a very good agreement between them.

Following the two-moons example, we examine the top three eigenvectors of L_s obtained by all the methods in Figs. 14 and 15. As it is difficult to directly visualize the distribution of the original 64-dimensional data in the x-y plane, we plot each eigenvector as a function of the row index and color each component according to the value of such index. Again, the row ordering of the input data X establishes the row correspondence to the eigenvector components. Similar to the two-moons case, the first eigenvector (first row), which is related to the normalized degree [26], is consistent between fully connected graph and all the Nyström approximations, while it is almost constant for KNN. Briefly, Figs. 14 and 15 illustrate that, aside from sign differences in the eigenvectors, both Nyström variants produce a good approximation to the first eigenvectors, while the KNN method produces different

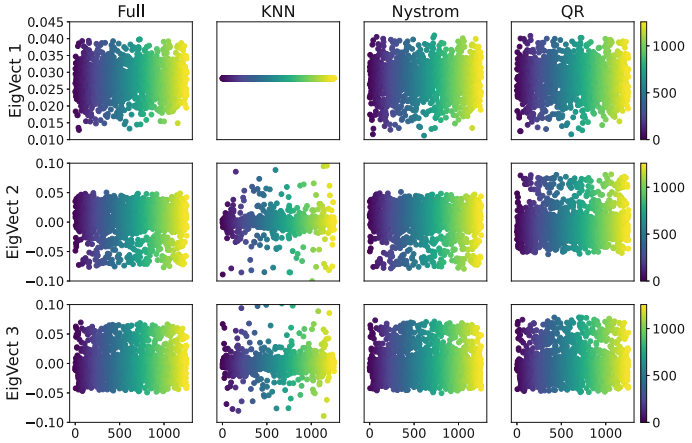

Fig. 14 Eigendecomposition of the symmetric normalized Laplacian for the digits dataset for each of the methods with $\sigma^2 = 1.0$, $K = 10$ for KNN, and $p = 250$ for Nyström methods

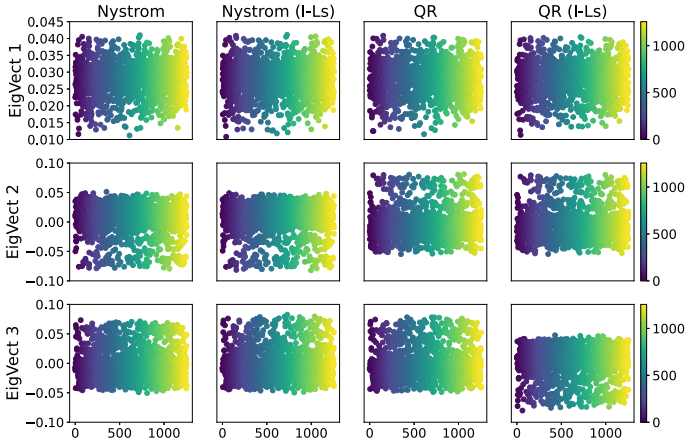

Fig. 15 Eigendecomposition of the symmetric normalized Laplacian obtained by Nyström methods ($p = 250$) with the two approximation variants on the digits dataset with $\sigma^2 = 1.0$

patterns. The errors in the approximations given by Eq. (13) are 0.071614 for KNN, 0.906414 for Nyström, 0.031120 for Nyström ($I - L_s$), 0.906467 for QR, and 0.030089 for QR ($I - L_s$). Similar to the two-moons case, from these error estimations, it is clear that the ($I - L_s$) variants of the Nyström methods produce much better approximations to the full symmetric normalized Laplacian than the direct L_s approximations.

We investigate the eigenvalues obtained by the competing methods under different values of σ^2; specifically, $\sigma^2 = 0.5, 1.0, 5.3$ and 10.3 are considered in

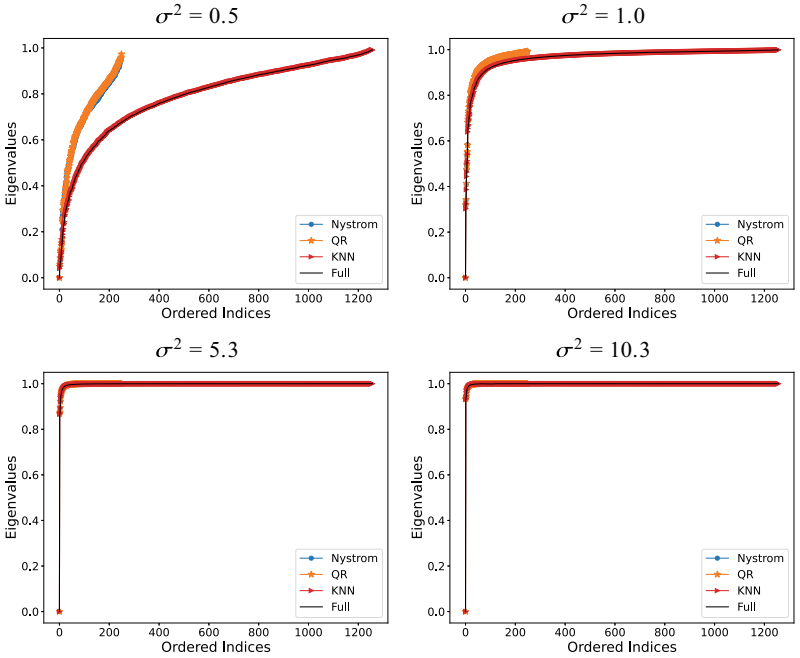

Fig. 16 Eigenvalues of the symmetric normalized Laplacian obtained by KNN ($K = 10$) and Nyström methods ($p = 250$) on the digits dataset under different values of σ^2. Note that Nyström methods completely overlap and only QR, which lays on top of original Nyström, is visible in the plots

Table 3 Comparison of the error E_r (13) of the approximation methods for the digits dataset for different σ^2 and for $K = 10$ (KNN) and $p = 250$ (Nyström methods)

	σ^2			
Method	0.5	1.0	5.3	10.3
KNN	0.337764	0.071614	0.007724	0.003838
Nyström	0.925103	0.906414	0.896262	0.895787
Nyström ($I - L_s$)	0.264682	0.031120	0.000317	0.000063
QR	0.922371	0.906467	0.896263	0.895788
QR($I - L_s$)	0.270744	0.030089	0.000296	0.000068

Fig. 16, showing that the smaller σ^2 is, the larger error to the fully connected graph is made by the Nyström approximations. However, in contrast with the two-moons case, for all these σ^2 values used, both the original Nyström and QR-based Nyström succeed. Table 3 records the approximation errors for these four values of σ^2. Note again that the $(I - L_s)$ variants yield small approximation errors.

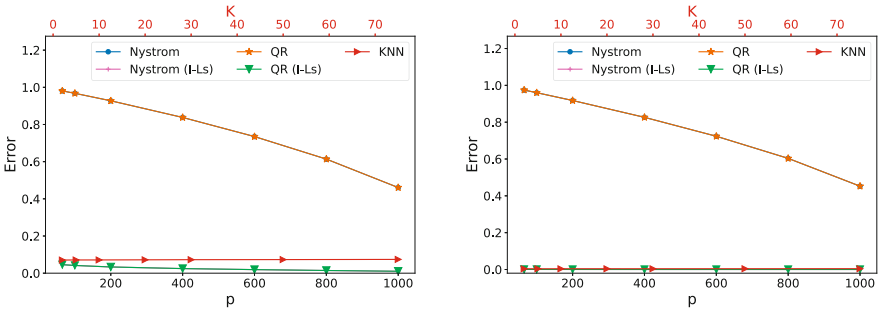

Fig. 17 Error in L_s approximation for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 1.0$ (left) and $\sigma^2 = 10.3$ (right), on the digits dataset. The results are averaged over 30 random trials and computed means are reported. The standard deviation computed is very small, with practically no-shaded region distinguishable. Note that Nyström methods completely overlap and only QR results, which fall on top of original Nyström , (or QR$(I - L_s)$ which fall on top of Nyström $(I - L_s)$), are visible in the plots

Approximation Errors

The approximation errors with respect to a range of K-nearest neighbors in KNN and p sampled data points in both Nyström methods are plotted in Fig. 17 for $\sigma^2 = 1.0$ and $\sigma^2 = 10.3$. The results are averaged over 30 random trials. Since the ranges of K and p are different, the plots include two x-axis: the top one in red corresponds to the K values for KNN, while the bottom one in black corresponds to the p values for Nyström methods. For this dataset, the Nyström method produces valid results across all the parameters tested. Figure 17 agrees with the observations made for the two-moons datasets, showing again that the approximations obtained via Nyström methods improve as the number of sample points p increases and that the error of the KNN method is smaller than the Nyström methods that directly approximate L_s and is relatively independent of K. Nyström methods that approximate $(I - L_s)$ produce smaller errors. The performance of Nyström methods on downstream tasks involving the eigendecomposition is better (see Fig. 19) or matches (see Fig. 20) the performance of the KNN method.

Computation Time

Under the same setup as the approximation error, the computation times are plotted in Fig. 18, where the standard deviations calculated over 30 random trials are depicted as a shaded region. As before, the times reported for KNN include the eigendecomposition stage. Figure 18 shows that the QR-based Nyström is slightly faster than the original Nyström method and that the KNN computation (via `giotto-tda` routine [25]) ensures stable computation times, remaining almost constant across the range $K \in [2, 75]$. Figure 18 reveals that there is a range

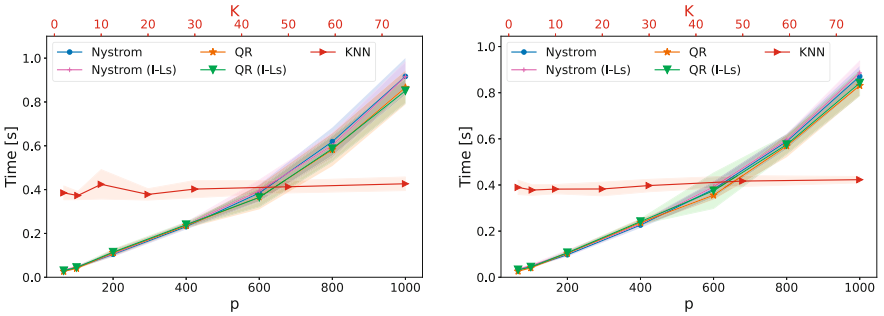

Fig. 18 Computation times for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 1.0$ (left) and $\sigma^2 = 10.3$ (right), on the digits dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials

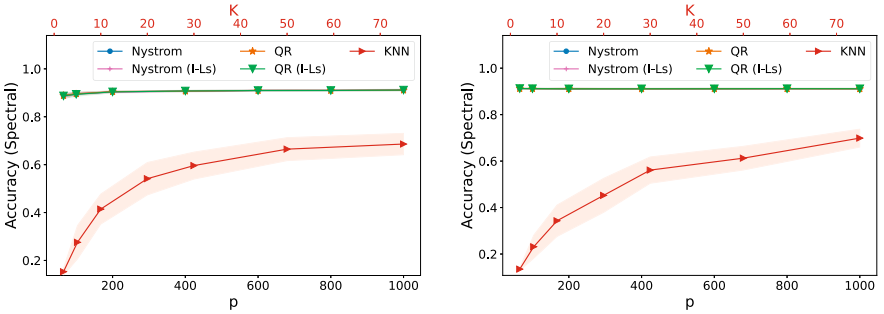

Fig. 19 Accuracy of spectral clustering for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 1.0$ (left) and $\sigma^2 = 10.3$ (right), on the digits dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials. Note that Nyström methods completely overlap and practically only QR($I - L_s$), which falls on top of the other Nyström variants, is visible in the plots

when substantial computation savings can be obtained by using the Nyström approximation methods, without a significant sacrifice in performance (see accuracy plots, e.g., Figs. 19 and 20).

Unsupervised Task

We report the performance of the weight approximation methods in the downstream task of unsupervised clustering. Averaged accuracy results obtained by spectral clustering over 30 random trials are plotted in Fig. 19 for $\sigma^2 = 1.0$ and $\sigma^2 = 10.3$. The standard deviations calculated over the random trials are depicted as a shaded region in the plots. Given that the eigenvectors tend to be more localized in KNN-

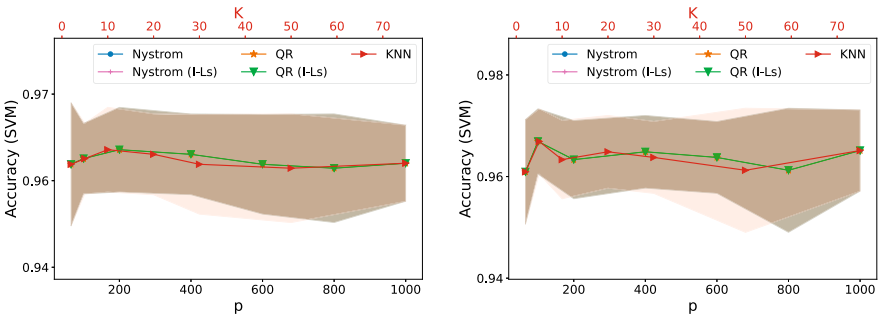

Fig. 20 Accuracy of SVM classification for KNN as a function of K (top axis) and Nyström methods as a function of p, for $\sigma^2 = 1.0$ (left) and $\sigma^2 = 10.3$ (right), on the digits dataset. The results are averaged over 30 random trials and computed means are reported. The shaded region in the plots represents the standard deviation calculated over the random trials. Note that Nyström methods completely overlap and only QR, which lays on top of original Nyström, is visible in the plots

based decompositions, 25 top eigenvectors were used for the spectral clustering, while only five top eigenvectors were used for Nyström methods. It is clear in Fig. 19 that projecting on the eigendecomposition of the Nyström methods produces good results, with around 90% accuracy, and these are much better than what is obtained with KNN. Nevertheless, in this case, major improvements in accuracy are observed for using a larger number of neighbors K in the KNN method.

Supervised Task

We also evaluate the downstream task of SVM classification and report results in Fig. 20. As observed before, the supervised learning improves the classification results, and again, even when the approximation to L_s computed by the Nyström methods has a larger error than KNN (see Fig. 17), the accuracy results are similar and deemed satisfactory in all cases.

3.2 CT Reconstruction

To test and compare the algorithms in different downstream processing tasks, we use a low-dose CT reconstruction problem with real image data of high dimensionality (256×256). In particular, we follow the MAGIC (manifold and graph integrative convolution network) approach [28], which unrolls a gradient descent algorithm into a neural network, using a convolutional neural network (CNN) to preserve pixel-level features and a graph convolutional network (GCN) to extract the nonlocal features from a patch-based manifold space. The graph is constructed by treating

every pixel of the CT image as a node and computing the weight using the Eq. (2) measured by the Euclidean distance between two small patches, whose top-left corner corresponds to the respective nodes. Then, the graph Laplacian is used in the GCN component of MAGIC to define the spectral graph convolution [5]. Here, the matrix composed of eigenvectors of the normalized graph Laplacian, i.e., V in Eq. (9), is analogous to the Fourier transform in standard spatial convolution, following the convolution theorem.

In what follows, we use three methods, KNN, Nyström, and QR-based Nyström, to approximate the computation of L_s for the GCN component of MAGIC and evaluate the obtained reconstructions in terms of peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and computational time. In all cases, we build the similarity matrix using a Gaussian similarity, Eq. (2), with $\sigma^2 = 5.7$. Note that given the high dimensionality of the data, we do not even attempt to build a fully connected graph for this case. We do not run Nyström variants that approximate $(I - L_s)$ since we expect similar performance to the one obtained with the direct L_s approximations. We follow the MAGIC work and use the same architecture and training parameters. For a proof of concept, we enact the following simplifications: (i) we use a reduced set of ten training images, (ii) we train for 50 epochs using a batch size of 2, and (iii) we test the trained model on ten test images different from the training set. We compare results for dose levels of 0.01 and 0.1 (see more details about the dose levels in the original work [28]).

Table 4 displays performance results for the reconstructions for the two dose levels or each of the three methods for computing L_s. The mean and standard deviations over the testing set are reported. Note that PSNR results are computed assuming a signal range in [0, 1], not the actual dynamic range. It can be observed that the results are very similar for all three methods, and of course, better results are obtained for measurements using a large dose level. Specific visual results are shown in Figs. 21 and 22 for dose levels of 0.01 and 0.1, respectively. Results for the lower-dose level have more granular artifacts, while results for the high-dose level are smoother (it may be necessary to zoom over the figures to note the difference). Finally, Fig. 23 shows a comparison of computation times on a GPU cluster (one node, eight NVIDIA GeForce RTX 2080 Ti GPUs), obtained for the three methods when approximating the symmetric normalized Laplacian for the coarse stage of the

Table 4 CT reconstruction comparison under two dose levels (0.01 and 0.1) for $K = 5$ (KNN) and $p = 50$ (Nyström methods)

		PSNR [dB]		SSIM	
Dose level	Method	Mean	Std	Mean	Std
0.01	KNN	35.60	0.38	0.9133	0.0066
	Nyström	35.60	0.38	0.9118	0.0063
	QR	36.04	0.39	0.9252	0.0057
0.10	KNN	41.36	0.36	0.9676	0.0033
	Nyström	41.33	0.37	0.9670	0.0033
	QR	41.13	0.38	0.9654	0.0036

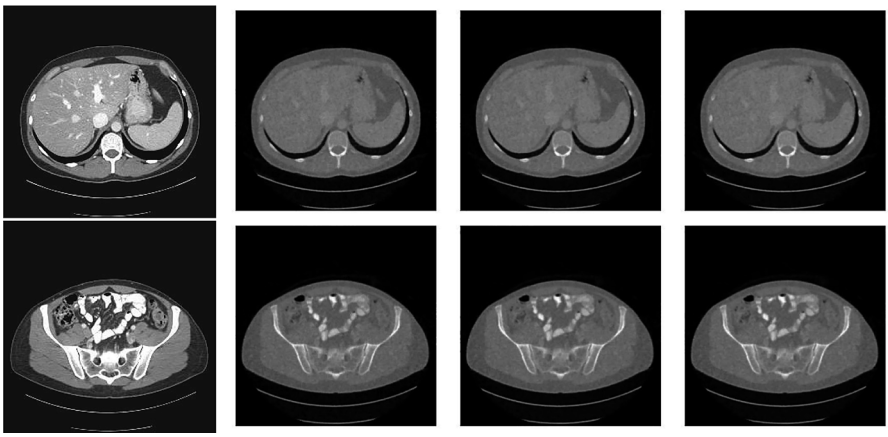

Fig. 21 Visual results of CT reconstruction under 0.01 dose level. From left to right: ground truth, KNN, Nyström, and QR

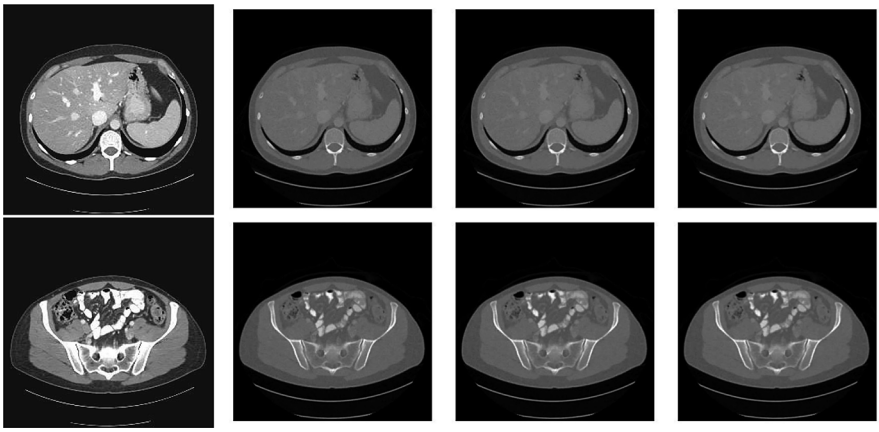

Fig. 22 Visual results of CT reconstruction under 0.1 dose level. From left to right: ground truth, KNN, Nyström, and QR

MAGIC reconstruction, using different numbers of p sampled data patches for the Nyström methods and different numbers of K patch neighbors for the KNN method. It is seen, consistent with results presented in previous sections, that the Nyström methods considerably reduce the computation time without significantly decreasing performance. Also, note that the QR-based Nyström method is slightly faster than the original Nyström method, which aligns with the observation in the synthetic case.

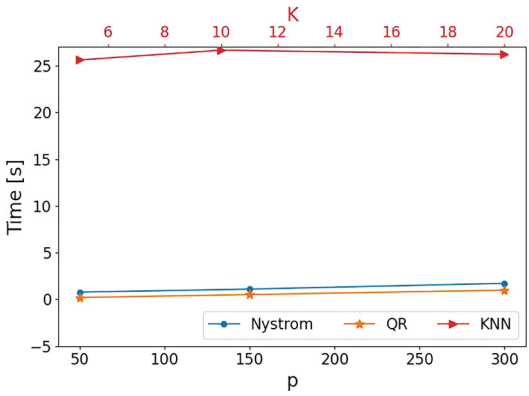

Fig. 23 Computation times for KNN as a function of K (top axis) and Nyström methods as a function of p when approximating the symmetric normalized Laplacian for the coarse stage of the MAGIC reconstruction

4 Conclusions

Through extensive numerical experimentation, including benchmarks as well as high-dimensional real datasets, we confirm the advantages of the Nyström methods for approximating the eigendecomposition of the symmetric Laplacian. Briefly, these methods provide accurate approximations of the eigenvalues and eigenvectors of a fully connected graph. Additionally, significant time savings are achieved by computing approximations based on eigendecompositions using subsets of data samples. The direct computation of eigenvalues and eigenvectors also facilitates the analysis of the graph structure, which is beneficial for downstream tasks such as clustering, classification, or graph-based signal filtering. We also observe that the QR method is slightly faster than the original Nyström method. However, the latter can become unstable or yield nonvalid solutions when a "large" number of data samples or a "large" value of σ^2 (resulting in the weight matrix being low rank) is used. It also seems the case that the Nyström approximations to the fully connected graph become worse when a "smaller" value of σ^2 is used. The problem, however, is that typically there is no a priori way to determine what "small" or "large" means in this context since it is heavily dataset-dependent. Overall, the QR-based method seems like a good alternative for more robust and faster approximations. Moreover, variants that approximate $(I - L_s)$ have much smaller approximation errors to the fully normalized symmetric Laplacian. It is also worth noticing that the relative Frobenius distance E_r can provide a somewhat misleading idea of the quality of the approximations, in particular when comparing the relative errors of KNN and Nyström methods. Although Nyström methods that directly approximate L_s seem to have worse errors compared to KNN and Nyström methods that approximate $(I - L_s)$ have much smaller approximation errors, their performance can be similar in downstream tasks.

Acknowledgments The authors would like to acknowledge the support from the Women in Data Science and Mathematics Research Workshop (WiSDM) hosted by IPAM at UCLA from August 7 to 11, 2023, which initiated the collaboration. C. Garcia-Cardona was funded by the Los Alamos

National Laboratory LDRD Program Director's Initiative (DI) project 20230771DI. Y. Liu is partially supported by NSF 2213436. Y. Lou is partially supported by NSF CAREER 2414705. S. Tang is partially supported by NSF 2111303 and NSF CAREER 2340631.

Competing Interests The authors have no conflicts of interest to declare that are relevant to the content of this chapter.

References

1. Alfke, D., Potts, D., Stoll, M., Volkmer, T.: NFFT meets Krylov methods: fast matrix-vector products for the graph Laplacian of fully connected networks. Front. Appl. Math. Stat. **4**, 61 (2018)
2. Bebendorf, M., Kunis, S.: Recompression techniques for adaptive cross approximation. J. Integral Equ. Appl. **21**(3), 331–357 (2009)
3. Belongie, S., Fowlkes, C., Chung, F., Malik, J.: Spectral partitioning with indefinite kernels using the nyström extension. In: Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part III 7, pp. 531–542. Springer, Berlin (2002)
4. Bertozzi, A.L., Flenner, A.: Diffuse interface models on graphs for classification of high dimensional data. Multiscale Model. Simul. **10**(3), 1090–1118 (2012)
5. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and deep locally connected networks on graphs (2014). arXiv preprint arXiv:1312.6203
6. Budd, J., van Gennip, Y., Latz, J.: Classification and image processing with a semi-discrete scheme for fidelity forced Allen–Cahn on graphs. GAMM-Mitteilungen **44**(1), e202100004 (2021)
7. Budd, J.M., van Gennip, Y., Latz, J., Parisotto, S., Schönlieb, C.B.: Joint reconstruction-segmentation on graphs. SIAM J. Imaging Sci. **16**(2), 911–947 (2023)
8. Chen, B., Lou, Y., Bertozzi, A.L., Chanussot, J.: Graph-based active learning for nearly blind hyperspectral unmixing. IEEE Trans. Geosci. Remote Sensing **61**, 1–16 (2023)
9. Chen, Y., Qi, L., Zhang, X.: The fiedler vector of a Laplacian tensor for hypergraph partitioning. SIAM J. Sci. Comput. **39**(6), A2508–A2537 (2017)
10. Cheng, X., Rachh, M., Steinerberger, S.: On the diffusion geometry of graph Laplacians and applications. Appl. Comput. Harmon. Anal. **46**(3), 674–688 (2019)
11. Chung, F.R.: Spectral Graph Theory, vol. 92. American Mathematical Society (1997)
12. Dong, W., Moses, C., Li, K.: Efficient k-nearest neighbor graph construction for generic similarity measures. In: Proceedings of the 20th International Conference on World Wide Web, pp. 577–586 (2011)
13. Dong, X., Thanou, D., Frossard, P., Vandergheynst, P.: Learning Laplacian matrix in smooth graph signal representations. IEEE Trans. Signal Process. **64**(23), 6160–6173 (2016)
14. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the Nyström method. IEEE Trans. Pattern Anal. Mach. Intell. **26**(2), 214–225 (2004)
15. Hart, R., Yu, L., Lou, Y., Chen, F.: Improvements on uncertainty quantification for node classification via distance based regularization. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
16. Hsu, C.W., Chang, C.C., Lin, C.J., et al.: A practical guide to support vector classification (2003)
17. McCollough, C.: TU-FG-207A-04: overview of the low dose ct grand challenge. Med. Phys. **43**(6), 3759–3760 (2016)
18. Merkurjev, E., Sunu, J., Bertozzi, A.L.: Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video. In: IEEE International Conference on Image Processing, pp. 689–693 (2014)

19. Ortega, A., Frossard, P., Kovačević, J., Moura, J.M., Vandergheynst, P.: Graph signal processing: overview, challenges, and applications. Proc. IEEE **106**(5), 808–828 (2018)
20. Ozaki, K., Shimbo, M., Komachi, M., Matsumoto, Y.: Using the mutual k-nearest neighbor graphs for semi-supervised classification on natural language data. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning, pp. 154–162 (2011)
21. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**(Oct), 2825–2830 (2011)
22. Perraudin, N., Paratte, J., Shuman, D., Martin, L., Kalofolias, V., Vandergheynst, P., Hammond, D.K.: Gspbox: a toolbox for signal processing on graphs (2014). ArXiv e-prints
23. Qin, J., Lee, H., Chi, J.T., Drumetz, L., Chanussot, J., Lou, Y., Bertozzi, A.L.: Blind hyperspectral unmixing based on graph total variation regularization. IEEE Trans. Geosci. Remote Sensing **59**(4), 3338–3351 (2020)
24. Shi, B., Han, L., Yan, H.: Adaptive clustering algorithm based on KNN and density. Pattern Recognit. Lett. **104**, 37–44 (2018)
25. Tauzin, G., Lupo, U., Tunstall, L., Pérez, J.B., Caorsi, M., Medina-Mardones, A., Dassatti, A., Hess, K.: giotto-tda: a topological data analysis toolkit for machine learning and data exploration (2020)
26. Von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**, 395–416 (2007)
27. Wu, Y., Ianakiev, K., Govindaraju, V.: Improved k-nearest neighbor classification. Pattern Recogn. **35**(10), 2311–2318 (2002)
28. Xia, W., Lu, Z., Huang, Y., Shi, Z., Liu, Y., Chen, H., Chen, Y., Zhou, J., Zhang, Y.: Magic: manifold and graph integrative convolutional network for low-dose ct reconstruction. IEEE Trans. Med. Imaging **40**(12), 3459–3472 (2021)
29. Zhang, B., Srihari, S.N.: Fast k-nearest neighbor classification using cluster-based trees. IEEE Trans. Pattern Anal. Mach. Intell. **26**(4), 525–528 (2004)