ABSTRACT

Diffusion maps are an emerging data-driven technique for non-linear dimensionality reduction, which are especially useful for the analysis of coherent structures and nonlinear embeddings of dynamical systems. However, the computational complexity of the diffusion maps algorithm scales with the number of observations. Thus, long time-series data presents a significant challenge for fast and efficient embedding. We propose integrating the Nyström method with diffusion maps in order to ease the computational demand. We achieve a speedup of roughly two to four times when approximating the dominant diffusion map components.

Index Terms— Dimension Reduction, Nyström method

2. DIFFUSION MAPS IN A NUTSHELL

Diffusion maps explore the relationship between heat diffusion and random walks on undirected graphs. A graph can be constructed from the data using a kernel function \( \kappa(x, y) : \mathcal{X} \times \mathcal{X} \to \mathbb{R} \), which measures the similarity for all points in the input space \( x, y \in \mathcal{X} \). A similarity measure is, in some sense, the inverse of a distance function, i.e., similar objects take large values. Therefore, different kernel functions capture distinct features of the data.

Given such a graph, the connectivity between two data points can be quantified in terms of the probability \( p(x, y) \) of jumping from \( x \) to \( y \). This is illustrated in Fig. 1. Specifically, the quantity \( p(x, y) \) is defined as the normalized kernel

\[
p(x, y) := \frac{\kappa(x, y)}{\nu(x)}.
\]

This is known as normalized graph Laplacian construction \([24]\), where \( \nu(x) \) is defined as a measure \( \nu(x) = \int_\mathcal{X} \kappa(x, y) \mu(y) \, dy \) of degree in a graph so that we have

\[
\int_\mathcal{X} p(x, y) \mu(y) \, dy = 1,
\]

where \( \mu(\cdot) \) denotes the measure of distribution of the data points on \( \mathcal{X} \). This means that \( p(x, y) \) represents the transition probability as a computational strategy and propose a Nyström-accelerated diffusion map algorithm.

Fig. 1: Nodes which have a high transition probability are considered to be highly connected. For instance, it is more likely to jump from node \( A \) to \( B \) than from \( A \) to \( F \).
tion kernel of a reversible Markov chain on the graph, i.e., $p(x, y)$ represents the one-step transition probability from $x$ to $y$. Now, a diffusion operator $P$ can be defined by integrating over all paths through the graph as

$$Pf(x) := \int p(x, y) f(y) \mu(y) \, dy, \quad \forall f \in L_1(\mathcal{X}), \quad (3)$$

so that $P$ defines the entire Markov chain [4]. More generally, we can define the probability of transition from each point to another by running the Markov chain $t$ times forward:

$$P^t f(x) := \int p^t(x, y) f(y) \mu(y) \, dy. \quad (4)$$

The rationale is that the underlying geometric structure of the dataset is revealed at a magnified scale by taking larger powers of $P$. Hence, the diffusion time $t$ acts as a scale, i.e., the transition probability between far away points is decreased with each time step $t$. Spectral methods can be used to characterize the properties of the Markov chain. To do so, however, we need to define first a symmetric operator $A$ as

$$A f(x) := \int a(x, y) f(y) \mu(y) \, dy \quad (5)$$

by normalizing the kernel with a symmetrized measure

$$a(x, y) := \frac{\kappa(x, y)}{\sqrt{\nu(x)} \sqrt{\nu(y)}}, \quad (6)$$

This ensures that $a(x, y)$ is symmetric, $a(x, y) = a(y, x)$, and positivity preserving $a(x, y) \geq 0 \ \forall x, y \ [5, 25]$. Now, the eigenvalues $\lambda_i$ and corresponding eigenvectors $\phi_i(x)$ of the operator $A$ can be used to describe the transition probability of the diffusion process. Specifically, we can define the components of the diffusion map $\Psi^t(x)$ as the scaled eigenfunctions of the diffusion operator

$$\Psi^t(x) = \left( \sqrt{\lambda_1^t} \phi_1(x), \sqrt{\lambda_2^t} \phi_2(x), ..., \sqrt{\lambda_n^t} \phi_n(x) \right).$$

The diffusion map $\Psi^t(x)$ captures the underlying geometry of the input data. Finally, to embed the data into an Euclidean space, we can use the diffusion map to evaluate the diffusion distance between two data points

$$D^2(x, y) = ||\Psi^t(x) - \Psi^t(y)||^2 \approx \sum_{i=1}^{d} \lambda_i^t (\phi_i(x) - \phi_i(y))^2,$$

where we may retain only the $d$ dominant components to achieve dimensionality reduction. The diffusion distance reflects the connectivity of the data, i.e., points which are characterized by a high transition probability are considered to be highly connected. This notion allows one to identify clusters in regions which are highly connected and which have a low probability of escape [3,5].

3. DIFFUSION MAPS MEET NYSTROEM

The Nyström method [26] provides a powerful framework to solve Fredholm integral equations which take the form

$$\int a(x, y) \phi_i(y) \mu(y) \, dy = \lambda_i \phi_i(x). \quad (7)$$

We recognize the resemblance with (5). Suppose, we are given a set of independent and identically distributed samples $\{x_1, x_j, ..., x_l\}$ drawn from $\mu(y)$. Then, the idea is to approximate Equation (7) by computing the empirical average

$$\frac{1}{l} \sum_{j=1}^{l} a(x, x_j) \phi_i(x_j) \approx \lambda_i \phi_i(x). \quad (8)$$

Drawing on these ideas, Williams and Seeger [22] proposed the Nyström method for the fast approximation of kernel matrices. This has led to a large body of research and we refer to [23] for an excellent and comprehensive treatment.

3.1. Nyström Accelerated Diffusion Maps Algorithm

Let us express the diffusion maps algorithm in matrix notation. Let $X \in \mathbb{R}^{n \times p}$ be a dataset with $n$ observations and $p$ variables. Then, given $\kappa$ we form a symmetric kernel matrix $K \in \mathbb{R}^{n \times n}$ where each entry is obtained as $K_{i,j} = \kappa(x_i, x_j)$.

The diffusion operator in Equation (3) can be expressed in the form of a diffusion matrix $P \in \mathbb{R}^{n \times n}$ as

$$P := D^{-\frac{1}{2}} K D^{-\frac{1}{2}}, \quad (10)$$

which allows us to compute the eigendecomposition

$$A = U \Lambda U^\top. \quad (11)$$

The columns $\phi_i \in \mathbb{R}^n$ of $U \in \mathbb{R}^{n \times n}$ are the orthonormal eigenvectors. The diagonal matrix $\Lambda \in \mathbb{R}^{n \times n}$ has the eigenvalues $\lambda_1 \geq \lambda_2 \geq ... \geq 0$ in descending order as its entries.

The Nyström method can now be used to quickly produce an approximation for the dominant $d$ eigenvalues and eigenvectors [22]. Assuming that $A \in \mathbb{R}^{n \times n}$ is a symmetric positive semidefinite matrix (SPSD), the Nyström method yields the following low-rank approximation for the diffusion matrix

$$A \approx C W^{-1} C^\top, \quad (12)$$

where $C$ is an $n \times d$ matrix which approximately captures the row and column space of $A$. The matrix $W$ has dimension $d \times d$ and is SPSD. Following, Halko et al. [12], we can factor $A$ in Equation (12) using the Cholesky decomposition

$$A \approx F F^\top, \quad (13)$$
where $F \in \mathbb{R}^{n \times d}$ is the approximate Cholesky factor, defined as $F := CW^{-1/2}$. Then, we can obtain the eigenvectors and eigenvalues by computing the singular value decomposition

$$F = \tilde{U}\Sigma V^\top. \quad (14)$$

The left singular vectors $\tilde{U} \in \mathbb{R}^{n \times d}$ are the dominant $d$ eigenvectors of $A$ and $\Lambda = \Sigma^2 \in \mathbb{R}^{d \times d}$ are the corresponding $d$ eigenvalues. Finally, we can recover the eigenvectors of the diffusion matrix $P$ as $U = D\tilde{U}$.

### 3.2. Matrix Sketching

Different strategies are available to form the matrices $C$ and $W$. Column sampling is most computational and memory efficient. Random projections have the advantage that they often provide an improved approximation. Thus, the different strategies pose a trade-off between speed and accuracy and the optimal choice depends on the specific application.

#### 3.2.1. Column Sampling

The most popular strategy to form $C \in \mathbb{R}^{n \times d}$ is column sampling, i.e., we sample $d$ columns from $A$. Subsequently, the small matrix $W \in \mathbb{R}^{d \times d}$ is formed by extracting $d$ rows from $C$. Given an index vector $J \in \mathbb{N}^n$ we form the matrices as

$$C := A(:, J) \quad \text{and} \quad W := C(J, :) = A(J, J). \quad (15)$$

The index vector can be designed using random (uniform) sampling or importance sampling [27]. Column sampling is most efficient, because it avoids explicit construction of the kernel matrix. For details we refer to [23].

#### 3.2.2. Random Projections

The second strategy is to use random projections [12]. First, we form a random test matrix $\Omega \in \mathbb{R}^{n \times l}$ which is used to sketch the diffusion matrix

$$S := A\Omega. \quad (16)$$

where $l \geq d$ is slightly larger than the desired target rank $d$. Due to symmetry, the columns of $S \in \mathbb{R}^{n \times l}$ provide a basis for both the column and row space of $A$. Then, an orthonormal basis $Q \in \mathbb{R}^{n \times l}$ is obtained by computing the QR decomposition as $S = QR$. We form the matrix $C \in \mathbb{R}^{n \times l}$ and $W \in \mathbb{R}^{l \times l}$ by projecting $A$ to a lower-dimensional space as

$$C := AQ \quad \text{and} \quad W := Q^\top C. \quad (17)$$

Further, the power iteration scheme can be used to improve the quality of the basis matrix $Q$ [12]. The idea is to sample from a preprocessed matrix $S = (A^2)^q A\Omega$, instead of directly sampling from $A$ as in Equation (16). Here, $q$ denotes the number of power iterations. In practice, this is implemented efficiently via subspace iterations.

### 4. RESULTS

In the following, we demonstrate the efficiency of our proposed Nyström accelerated diffusion map algorithm. First, we explore both toy data and time-series data from a dynamical system. Then, we evaluate the computational performance and compare it with the deterministic diffusion map algorithm. Here, we restrict the evaluation to the Gaussian kernel:

$$\kappa(x, y) = \exp \left( -\sigma^{-1} ||x - y||_2^2 \right),$$

where $\sigma$ controls the variance (width) of the distribution.

#### 4.1. Artificial Toy Datasets

First, we consider two non-linear artificial datasets: the helix and the famous Swiss role dataset. Both datasets are perturbed with a small amount of white Gaussian noise. Figure 2 shows both datasets. The first two components of the diffusion map $\Psi^t(x)$ are used to illustrate the non-linear embedding in two dimensional space at time $t = 100$. Then, we use the diffusion distance to cluster the data points. Indeed, the diffusion map is able to correctly cluster both non-linear data sets. The width of the Gaussian kernel is set to $\sigma = 0.5$.

**Fig. 2:** The top row shows the datasets. The second row shows the clustered data points at diffusion time $t = 100$. The third row shows low-dimensional embedding computed using the Nyström-accelerated diffusion map algorithm.
Fig. 3: The chaotic Lorenz system and its two-dimensional embedding using diffusion maps. Here, a large number of diffusion time steps $t$ is required to obtain stable clusters.

4.2. The Chaotic Lorenz System

Next, we explore the embedding of nonlinear time-series data. Discovering nonlinear transformations that map dynamical systems into a new coordinate system with favorable properties is at the center of modern efforts in data-driven dynamics. One such favorable transformation is obtained by eigenfunctions of the Koopman operator, which provides an infinite-dimensional but linear representation of nonlinear dynamical systems [28,30]. Diffusion maps have recently been connected to Koopman analysis and are now increasingly being employed to analyze coherent structures and nonlinear embeddings of dynamical systems [31–33]. Here, we explore the chaotic Lorenz system, which is among the simplest and well-studied chaotic dynamical system [34]:

$$\dot{x}, \dot{y}, \dot{z} = [\sigma(y - x), x(\rho - z) - y, xy - \beta z],$$

with parameters $\sigma = 10, \rho = 28,$ and $\beta = 8/3$. We use the initial conditions $[-8 \ 8 \ 27]^T$ and integrate the system from $t=0$ to $t=5$ with $\Delta t \approx 0.0001$. Figure 3 shows the results.

4.3. Computational Performance

Table 1 gives a flavor of the computational performance of the Nyström-accelerated diffusion map algorithm. We achieve substantial computational savings over the deterministic diffusion map algorithm, while attaining small errors. The relative errors between the deterministic $\Psi^t(x)$ and randomized diffusion maps $\tilde{\Psi}^t(x)$ at $t=1$ are computed in the Frobenius norm: $\|\|\Psi^t(x) - \tilde{\Psi}^t(x)\|_F/\|\Psi^t(x)\|_F$.

The algorithms are implemented in Python and code is available via GitHub: [https://github.com/erichson](https://github.com/erichson). The deterministic algorithm uses the fast ARPACK eigen-solver provided by SciPy. The Nyström accelerated diffusion map algorithm is computed using random projections with slight oversampling and two additional power iterations $q=2$. The target-rank (number of components) is set to $d = 300$ for the toy data and $d = 500$ for the Lorenz system. Figure 4 shows the approximated eigenvalues for the Lorenz system.

5. DISCUSSION

The computational complexity of diffusion maps scales with the number of observations $n$. Thus, applications such as the analysis of time-series data from dynamical systems pose a computational challenge for diffusion maps. Fortunately, the Nyström method can be used to ease the computational demands. However, diffusion maps are highly sensitive to the approximated range subspace which is provided by the eigenvectors. This means that the Nyström method provides a good approximation only if: (a) the kernel matrix has low-rank structure; (b) the eigenvalue spectrum has a fast decay. The Nyström method shows an excellent performance using random projections with additional power iterations. We achieve a speedup of roughly two to four times when approximating the dominant diffusion map components. Unfortunately, the approximation quality turns out be poor using random column sampling. Future research opens room for a more comprehensive evaluation study. Further, it is of interest to explore kernel functions which are more suitable for dynamical systems, e.g., cone-shaped kernels [31,35].
6. REFERENCES

[1] C. Burges, “Dimension reduction: A guided tour,” Foundations and Trends in Machine Learning, vol. 2, no. 4, pp. 275–365, 2010.

[2] L. Van Der Maaten, E. Postma, and J. Van den Herik, “Dimensionality reduction: A comparative review,” Journal of Machine Learning Research, vol. 10, pp. 66–71, 2009.

[3] S. Lafon, Diffusion maps and geometric harmonics, Ph.D. thesis, Yale University PhD dissertation, 2004.

[4] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker, “Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps,” Proceedings of the National Academy of Sciences, vol. 102, no. 21, pp. 7426–7431, 2005.

[5] R. R. Coifman and S. Lafon, “Diffusion maps,” Applied and Computational Harmonic Analysis, vol. 21, no. 1, pp. 5–30, 2006.

[6] R. R. Coifman, I. G. Kevrekidis, S. Lafon, M. Maggioni, and B. Nadler, “Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems,” Multiscale Modeling & Simulation, vol. 7, no. 2, pp. 842–864, 2008.

[7] B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis, “Diffusion maps, spectral clustering and reaction coordinates of dynamical systems,” Applied and Computational Harmonic Analysis, vol. 21, no. 1, pp. 113–127, 2006.

[8] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, “Fast high dimensional vector multiplication face recognition,” in Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 1960–1967.

[9] L. Karacan, E. Erdem, and A. Erdem, “Structure-preserving image smoothing via region covariances,” ACM Trans. Graph., vol. 32, no. 6, pp. 1–11, 2013.

[10] R. Xu, S. Damelin, and D. C. Wunsch, “Applications of diffusion maps in gene expression data-based cancer diagnosis analysis,” in Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE, IEEE, 2007, pp. 4613–4616.

[11] G. Mishne and I. Cohen, “Multiscale anomaly detection using diffusion maps,” IEEE Journal of Selected Topics in Signal Processing, vol. 7, no. 1, pp. 111–123, 2013.

[12] N. Halko, P. Martinsson, and J. A. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions,” SIAM Review, vol. 53, no. 2, pp. 217–288, 2011.

[13] M. W. Mahoney, “Randomized algorithms for matrices and data,” Foundations and Trends in Machine Learning, vol. 3, no. 2, pp. 123–224, 2011.

[14] E. Liberty, “Simple and deterministic matrix sketching,” in Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2013, pp. 581–588.

[15] N. B. Erichson, S. Voronin, S. L. Brunton, and J. N. Kutz, “Randomized matrix decompositions using R,” arXiv preprint arXiv:1608.02148, 2016.

[16] N Benjamin Erichson, Steven L Brunton, and J Nathan Kutz, “Compressed singular value decomposition for image and video processing,” in 2017 IEEE International Conference on Computer Vision Workshop (ICCVW). IEEE, 2017, pp. 1880–1888.

[17] N. Halko, P. Martinsson, Y. Shkolnisky, and M. Tygert, “An algorithm for the principal component analysis of large data sets,” SIAM Journal on Scientific Computing, vol. 33, no. 5, pp. 2580–2594, 2011.

[18] V. Rohklin, A. Szlam, and M. Tygert, “A randomized algorithm for principal component analysis,” SIAM Journal on Matrix Analysis and Applications, vol. 31, no. 3, pp. 1100–1124, 2009.

[19] N. B. Erichson, A. Mendible, S. Wihlborn, and J. N. Kutz, “Randomized nonnegative matrix factorization,” Pattern Recognition Letters, vol. 104, pp. 1–7, 2018.

[20] N. B. Erichson, S. L. Brunton, and J. N. Kutz, “Randomized dynamic mode decomposition,” arXiv preprint arXiv:1702.02912, 2017.

[21] N Benjamin Erichson, Krithika Manohar, Steven L Brunton, and J Nathan Kutz, “Randomized cp tensor decomposition,” arXiv preprint arXiv:1703.09074, 2017.

[22] C. Williams and M. Seeger, “Using the Nyström method to speed up kernel machines,” in Advances in Neural Information Processing Systems 13, pp. 682–688, 2001.

[23] B. O. Koopman, “Hamiltonian systems and transformation in dynamical systems,” in Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, no. 1-3, pp. 309–325, 2005.

[24] B. O. Koopman, “Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control,” PLoS ONE, vol. 11, no. 2, pp. e0150171, 2016.

[25] D. Giannakis, “Dynamics-adapted cone kernels,” SIAM Journal on Applied Dynamical Systems, vol. 14, no. 2, pp. 556–608, 2015.