Manifold Alignment Aware Ants: A Markovian Process for Manifold Extraction

Mohammad Mohammadi
mohammadimathstar@gmail.com
Faculty of Science and Engineering, University of Groningen, Groningen 9747AA, The Netherlands

Peter Tino
p.tino@cs.bham.ac.uk
Department of Computer Science, University of Birmingham, Birmingham B15 2TT

Kerstin Bunte
k.bunte@rug.nl
Faculty of Science and Engineering, University of Groningen, Groningen 9747 AG, The Netherlands

The presence of manifolds is a common assumption in many applications, including astronomy and computer vision. For instance, in astronomy, low-dimensional stellar structures, such as streams, shells, and globular clusters, can be found in the neighborhood of big galaxies such as the Milky Way. Since these structures are often buried in very large data sets, an algorithm, which can not only recover the manifold but also remove the background noise (or outliers), is highly desirable. While other works try to recover manifolds either by pushing all points toward manifolds or by downsampling from dense regions, aiming to solve one of the problems, they generally fail to suppress the noise on manifolds and remove background noise simultaneously. Inspired by the collective behavior of biological ants in food-seeking process, we propose a new algorithm that employs several random walkers equipped with a local alignment measure to detect and denoise manifolds. During the walking process, the agents release pheromone on data points, which reinforces future movements. Over time the pheromone concentrates on the manifolds, while it fades in the background noise due to an evaporation procedure. We use the Markov chain (MC) framework to provide a theoretical analysis of the convergence of the algorithm and its performance. Moreover, an empirical analysis, based on synthetic and real-world data sets, is provided to demonstrate its applicability in different areas, such as improving the performance of *t*-distributed stochastic neighbor embedding (t-SNE) and spectral clustering using the underlying MC formulas,

recovering astronomical low-dimensional structures, and improving the performance of the fast Parzen window density estimator.

1 Introduction

New technological developments facilitate the collection of large amounts of high-dimensional data in different fields, such as astronomy, sensor networks, medical science, and computer vision. A typical challenge in dealing with high-dimensional data is the curse of dimensionality, where the data space is sparse, such that data points are far from their neighbors. However, in practice, the high-dimensional data is often generated by a system governed by a small number of underlying components (Dixit, 2019). In other words, the high-dimensional data lie on a lower-dimensional topological structure called a manifold. Therefore, many dimensionality reduction methods (e.g., Roweis & Saul, 2000; Belkin & Niyogi, 2003; Donoho & Grimes, 2003; Zhang & Zha, 2003; Coifman & Lafon, 2006) aim to identify this underlying manifold from the data. However, such approaches might be tremendously impeded in their performance because of the presence of noise and outliers (Wang & Carreira-Perpinán, 2010). This led to the development of methods aiming to deal with noisy manifolds and outliers.

There are different strategies to decrease the impact of noise on manifold learning algorithms. The technique proposed in Little, Maggioni, and Murphy (2020) denoises data sets via downsampling and picking samples in high-dense regions, whereas others suppress the noise level via pushing all instances toward regions with higher density. While the first category can deal with background noise (or outliers not belonging to any manifold), the basic assumption in the second category is that the noisy samples belong to a manifold, and their deviations from the manifold are caused by measurement noise. An example technique is manifold denoising (MD) (Hein & Maier, 2006), where the denoising process is modeled as a diffusion process on a neighborhood graph using the Laplacian to denoise the manifold. The aim of Klicpera, Weissenberger, and Günnemann (2019) is to generalize the notion of diffusion operator on graphs so that larger node neighborhoods than one-hop in the neighborhood graph are considered. To this end, the authors first construct a transition graph (that can be renormalized to define a MC) that encompasses a variety of (potentially unbounded) node neighborhoods, albeit strongly downweighting increasingly large ones. The resulting dense transition graph then needs to be sparsified to consider only (potentially) higher-order neighborhoods that "really matter" (i.e., have strong support in the transition structure). The generalized diffusion convolution (GDC) is then formulated in such a new neighborhood structure. Wang and Carreira-Perpinán (2010) propose manifold blurring mean shift (MBMS), where the mean shift directions are only allowed to be parallel to manifold normals. Although increasing the number of iterations in the above algorithms may cause a better result, they eventually partition the manifold into local clusters. To overcome this problem, locally linear denoising (LLD) (Gong, Sha, & Medioni, 2010) proposes a noniterative method, assuming that a manifold can be explained as a set of overlapping linear patches. While it denoises each patch separately, it simultaneously uses the graph Laplacian to achieve a smooth manifold. However, such denoising methods may fail in applications, such as astronomy, where noisy manifolds are buried inside point clouds. Thus, a method capable of handling both types of noise, along with a manifold and background, is highly desirable.

Natural systems, such as ant colonies, have motivated many swarm algorithms in computer science. For instance, the cooperation among ants in the food-seeking process has inspired many solutions for optimization problems (Dorigo, Maniezzo, & Colorni, 1991, 1996; Dorigo, 1992; Stützle & Hoos, 2000; Maniezzo, 1999; Blum, Roli, & Dorigo, 2001) and clustering (Tsai, Tsai, Wu, & Yang, 2004; Chu, Roddick, Su, & Pan, 2004; Runkler, 2005). In nature, when an ant leaves its nest to find food, it deposits a chemical substance called pheromone on its path, which serves as information for other ants that are attracted to it. Each ant's decision is influenced by the pheromone: the higher the concentration is on a particular route, the more likely it is chosen as a path to follow. This seemingly uncoordinated local behavior in collection assists the hive to find the shortest path to the food source. Following this strategy, several algorithms were proposed and applied to optimization problems. Their simplicity and flexibility make these methods desirable for many optimization tasks that include graphs, such as vehicle/Internet routing (Rizzoli, Oliverio, Montemanni, & Gambardella, 2004) and water distribution systems (Gil et al., 2011).

Unfortunately, the strategies explained above fail to denoise data comprising noisy manifolds contained in point clouds, which are often faced in application domains involving particle simulations, such as astronomy. In this contribution, we propose a new algorithm to uncover noisy manifolds buried in high-dimensional background noise. Motivated by ant colony optimization strategies, our method employs multiple artificial ants that jump from point to point based on defined preferences and release pheromone. In previous work we proposed a strategy in which ants were allowed to transport data points toward the manifolds and hence denoising and "cleaning" them (Mohammadi & Bunte, 2020). In contrast, this contribution aims to detect noisy manifolds in potentially large amounts of background noise. Here, the position of data points is not changed; instead, the ants prefer to move toward low-dimensional structures and deposit pheromone on the points they visit. The latter serves as positive feedback that accumulates and concentrates on the manifolds, leaving the noisy points with relatively less pheromone and hence highlighting the structures. Both our approach and GDC are built on a notion of a stochastic transition matrix reflecting the connection structure of a given graph. While in our case, the neighborhood graph represents local geometric structures of a point cloud, in GDC, pairwise relationships are used (even though in applications, the graphs do often represent point clouds). In general, our approach differs from graph diffusion (GD) in several important aspects:

1. In our approach, the adjacency structure of the neighborhood graph by itself is not sufficient. We explicitly enforce a preference for neighbors that are aligned with low-dimensional structures in the point cloud the neighborhood graph represents.
2. Motivated by ant colony algorithms, we allow for an additional mechanism of positive reinforcement when discovering low-dimensional point structures in a noisy background, namely, pheromone accumulation and evaporation.
3. We do not perform any graph convolution; instead, we associate the "importance values" of each node with the amount of accumulated pheromone. Note that these values potentially reflect large-scale geometric structures as the pheromone is deposited over long-range ant walks. In this sense, our approach shares the ambition of GDC for representing larger-scale "important" neighborhoods. However, in GDC, the larger-scale structures are rapidly weighted down, whereas in our case, the pheromone deposits can survive over large-scale structures, if frequently visited by the ants.

With empirical experiments, we demonstrate the ability of our algorithm to highlight manifolds in synthetic and real-world applications. Furthermore, we show that the pheromone distribution can be used to improve the performance of the fast Parzen window density estimator. In addition to an empirical analysis, we provide a thorough theoretical analysis to verify the ability of pheromone to encode information about distances of data points to the underlying manifold. First, we only focus on a linear manifold and use the Markov chain framework to study the behavior of the algorithm. In order to extend our analysis to nonlinear manifolds, we assume that a manifold can be approximated by locally linear patches, and we analyze how the pheromone distribution asymptotically behaves. Our analysis shows that the pheromone sorts data points according to their distances to the linear patch: the closer a data point is to the linear patch, the higher pheromone level it has. In summary, we propose an algorithm that provides valuable information, called pheromone, about how far points are from the underlying manifolds. This information is beneficial not only to extract noisy manifolds buried in point clouds, but also to build more effective density estimators.

The organization of this article is as follows. In section 2, we provide background information to study random walks in general. Section 3 contains three alternative transition probabilities for random walks, and our proposed method to extract manifolds from a point cloud is exhibited in section 4. A theoretical analysis of the role of the pheromone and its convergence is presented in section 5. Section 6 contains an empirical analysis on synthetic data sets and real-world application examples, demonstrating the performance in recovering manifolds and improving visualization and density estimation techniques.

2 Background and Notation

Let $X = [x_1 \quad x_2 \quad \cdots \quad x_N] \in \mathbb{R}^{D\times N}$ be the data matrix storing N D-dimensional data points in its columns. Given a number $r > 0$, we define the neighborhood of any point $\boldsymbol{x}_i$ as

$$B_r(\boldsymbol{x}_i) = \{\boldsymbol{x}_j |\ \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 < r, 1 \le j \le N\}, \tag{2.1}$$

where $\|.\|$ denotes the Euclidean norm. This definition of neighborhood has an advantage of being geometrically motivated (Belkin & Niyogi, 2003), which also inspires our theoretical analysis. Alternatives are methods such as k-nearest neighbor and mutual k-nearest neighbor (Von Luxburg, 2007). In the neighborhood, graph nodes correspond to the data points from X, and $(\boldsymbol{x}_i, \boldsymbol{x}_k)$ represents the directed edge connecting $\boldsymbol{x}_i$ to $\boldsymbol{x}_k$, with the associated weight $w(\boldsymbol{x}_i, \boldsymbol{x}_k) \ge 0$. The usual way to describe a Markov chain (MC) is to use a weighted graph where the nodes denote the states of the MC, and its weights represent the probability of transitions between the states. Thus, from the above neighborhood graph, one can construct an MC by row normalization,

$$p_{ij} = \frac{w(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\sum_{k\in B_r(\boldsymbol{x}_i)} w(\boldsymbol{x}_i, \boldsymbol{x}_k)}, \tag{2.2}$$

where p_{ij} is the transition probability of jumping from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$. A popular MC process is the random walk where a walker follows the transition probabilities to move on the graph. We now review some of the key results and notions from the MC theory that will be needed in our study.

Assume $S = \{1, 2, \ldots, N\}$ represents the state space of a system. Let X_n denote the random variable of the state of the system in the nth time step. In an MC, we assume that the next state of the system only depends on the current state, and the system's behavior is described through transition probabilities,

$$p_{ij} = P(X_1 = j | X_0 = i), \qquad \forall i, j \in S, \tag{2.3}$$

with X_0 being the initial state of the system. In this section, we only consider homogeneous MCs where the transition probabilities are independent of the time n. Now, let $P = [p_{ij}]$ be a matrix containing the transition probabilities. While P includes the transition probabilities for one time step, it can be extended to an arbitrary number of steps:

Theorem 1. *For a MC with the transition probability matrix P, we have*

$$P(X_n = j|X_0 = i) = p_{ij}^{(n)}$$

where $p_{ij}^{(n)}$ *is* (i, j)*th element of the* n*th power of the matrix* P*, that is,* $P^n = \underbrace{P \times \ldots \times P}_{n \text{ times}}$ *(Kulkarni, 1999).*

This distribution is used to study an MC in a limited time interval. Let $N_j(n)$ denote the number of visits state j accumulates over $(n+1)$ steps, including the initial state. To study the behavior of a chain in this interval the occupancy time is defined as follows:

Definition 1. *For any state* j *and initial state* i*, the occupancy time of state* j *up to* n *is the expected number of times the system spends in state* j *in a random walk of* $n+1$ *steps starting in* i*,*

$$m_{ij}(n) = E[N_j(n)|X_0 = i],$$

where $E[\cdot]$ *denotes the expectation operator.*

Let $M(n) = [m_{ij}(n)]$ denote the occupancy time matrix. The following theorem connects the occupancy time to the power of transition matrix (Kulkarni, 1999):

Theorem 2. *Given an MC, its occupancy time matrix is*

$$M(n) = \sum_{q=0}^{n} P^q. \tag{2.4}$$

While this theorem helps to study an MC in a finite number of time steps, we would like to investigate the long-term behavior of the chain, which is encoded in the stationary distribution:

Definition 2. *For an MC with the transition probability matrix P, the distribution* $\pi = [\pi_1, \ldots, \pi_N]$ *is called its stationary distribution if the following balance equation holds:*

$$\pi P = \pi. \tag{2.5}$$

Here, we only focus on a specific class of Markov chains called *ergodic chains*:

Definition 3. *An MC is called ergodic if the following condition holds:*

$$\exists T \in \mathbb{N} : \forall n > T, \forall i, j \in S \quad P(X_n = j | X_0 = i) > 0.$$

In other words, for sufficiently large time steps n, the system can be found in any state. The following theorem shows that the stationary distribution captures all information about the long-term behavior of an ergodic MC (Cinlar, 2013; Kulkarni, 1999):

Theorem 3. *For an ergodic MC its stationary distribution π is unique with:*

$$\forall i \in S; \ \lim_{n\to\infty} P(X_n = j|X_0 = i) = \pi_j, \tag{2.6}$$

$$\lim_{n\to\infty} P(X_n = j) = \pi_j, \tag{2.7}$$

and the expected return time to state j is $\mu_j = \dfrac{1}{\pi_j}$. (2.8)

3 Homogeneous Markov Chain

An important class of manifold learning techniques relies on weighted graph representations of data. Markov chains constructed on graphs offer a powerful tool to represent nonlinear manifolds and are used in several dimensionality-reduction techniques. This is exemplified by a well-known family of dimensionality-reduction techniques called "kernel eigenmap methods," which includes local linear embedding (Roweis & Saul, 2000), Laplacian eigenmaps (Belkin & Niyogi, 2003), Hessian eigenmaps (Donoho & Grimes, 2003), local tangent space alignment (Zhang & Zha, 2003), and diffusion map (Coifman & Lafon, 2006). The general idea is that the eigenvectors of Markov transition matrices are used to project high-dimensional data to a lower-dimensional Euclidean space preserving the main structures of the data (Coifman & Lafon, 2006). Therefore, applying MCs in the context of manifold learning has been employed in several techniques.

Although manifold learning techniques assume that the data lie on a lower-dimensional topological structure, in practice, it rarely is clear due to the presence of noise. One way to improve their performance is to use "less noisy data" closely aligned with the underlying manifold. In this contribution, the goal is to perform a random walk that places emphasis on such manifold aligned subsamples. In the following, we introduce three MCs that can recover a manifold by highlighting sample points closer to it.

3.1 Weights Based on Kernels.

A common way to construct an MC on a data set is to use kernel functions (Berry & Sauer, 2016), that is, a map $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ that fulfills

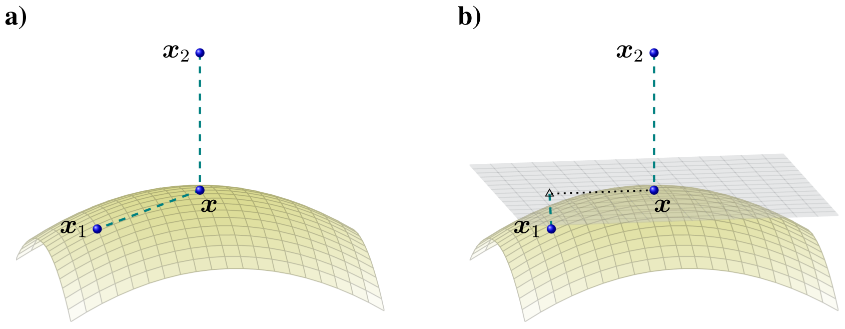

Figure 1: An illustration of a curved manifold. In contrast to x_2, the points x and x_1 lie on the curved manifold. The Euclidean distance (dashed) to both x_1 and x_2 is equal (see panel a). As shown in panel b, the distance to the tangent (dashed) reflects the closeness to the manifold much better.

- $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D \quad K(\boldsymbol{x}, \boldsymbol{y}) = K(\boldsymbol{y}, \boldsymbol{x})$
- $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D \quad K(\boldsymbol{x}, \boldsymbol{y}) \geq 0$

and quantifies a "similarity" between pairs of data points. The most popular kernel function is the gaussian, which for two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as

$$K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right), \tag{3.1}$$

with the Euclidean distance $\|.\|$ and scale parameter $\sigma > 0$. Based on the gaussian kernel, the transition probability reads

$$p_{ij} = \frac{K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\sum_{k \in \mathcal{N}_i} K_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_k)},$$

with $\mathcal{N}_i$ being the set of point $\boldsymbol{x}_i$'s neighbors. This definition of transition probabilities reinforces random walkers to spend more time in denser regions.

3.2 Weights Based on Tangent Spaces. Assuming the presence of a manifold, the noise level of a data point can be related to its distance from the manifold. Thus, to form an MC that encourages random walkers to spend more time on the manifold, the walkers need to know how far data points are from the manifold. As depicted in Figure 1a, the Euclidean distance fails to reveal this information. There, if a random walker resides on point $\boldsymbol{x}$, the gaussian kernel gives the walker the same chance to stay on the manifold (point x_1) or leave it (jumping to point x_2).[1] In order to overcome this drawback, we need to define a favorable measure that improves the approximate distances to the manifold.

A manifold can be approximated locally at a point x by its tangent space (Tu, 2011). Hence, we propose to use the tangent space to estimate the distance of x's neighbors to the manifold (see Figure 1b). Let $\mathcal{N}$ represent the set of x's neighbors, for example, enclosed in $B_r(x)$. The tangent space at x is typically approximated by principal component analysis (PCA) on $\mathcal{N}$[2], yielding a set of eigenvalues and unit orthogonal eigenvectors of the local covariance matrix, $\{(\lambda_k, u_k)\}_{k=1}^{D}$. Without loss of generality, we assume that the eigenvalues are in descending order with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_D \geq 0$ and normalized to $\sum_k \lambda_k = 1$. Then for a d-dimensional manifold, the subspace spanned by the columns of $U_1 = \left[u_1, u_2, \ldots, u_d\right]$ provides an estimation for the tangent space and locally for the manifold.

Since we consider noisy data sets we only have access to a noisy version of the tangent space $\hat{U}_1$. Hence, depending on the noise level, the difference between U_1 and $\hat{U}_1$ may be small or high. For more details, see Kaslovsky and Meyer (2014) and Little, Maggioni, and Rosasco (2017). In the following we introduce two ways to define MCs highlighting samples close to manifold structures.

3.2.1 A Manifold with Known Intrinsic Dimensionality. Consider a d-dimensional manifold in x's neighborhood. For any $x_i \in \mathcal{N}$, one can estimate the distance to the tangent space by

$$\delta_i^{\mathcal{M}} = \|\left(I - \hat{U}_1\hat{U}_1^T\right)(x_i - x')\|, \tag{3.2}$$

where x' determines the place where the tangent space touches the manifold in the neighborhood. Based on these quantities, we define the weight values to be used in the calculation of transition probabilities equation 2.2 as follows:

$$w_d(x, x_i) = \begin{cases} 1 - \frac{\delta_i^{\mathcal{M}}}{\alpha} & \text{if } \alpha \geq \delta_i^{\mathcal{M}} \\ 0 & \text{if } \alpha < \delta_i^{\mathcal{M}}, \end{cases} \tag{3.3}$$

where α is a factor to ensure the weight values are positive. Here, we determine α such that only p percent of neighbors have nonzero weights. This

[1] A similar problem holds for the GDC, introduced in Klicpera et al. (2019). Since it enforces symmetric weights between every point pair, it uses a similar weight for jumping from x to x_2 and vice versa.

[2] Note that methods, such as Lerman, McCoy, Tropp, and Zhang (2015), provide better estimation for the tangent spaces. However, since in this contribution we consider big data sets, we use the less costly PCA.

definition constructs a random walk favoring jumps closer to the manifold. In other words, the random walker observes a flat region and is more likely to move to a point close to it.

3.2.2 A Manifold with Unknown Intrinsic Dimensionality. In most applications, the dimensionality of a manifold d is unknown, and eigenvalues are typically used to estimate it. As suggested in Wang, Tino, and Fardal (2008), the intrinsic dimensionality may be estimated by

$$\hat{d} = \arg\max_{d} S_d,$$

where $S_d = d \cdot (\lambda_d - \lambda_{d+1})$, $d \in \{1, \dots, D-1\}$, and $S_D = D \cdot \lambda_D$. However, the performance of these types of criteria highly depends on the neighborhood size, which is related to the noise level and the manifold curvature. In order to tackle this problem, we consider all possible values of d instead of picking a specific value. The basic idea is to compute the weight values for any $d \in \{1, \dots, D\}$, according to equation 3.3, followed by the calculation of their mean,

$$w(\boldsymbol{x}, \boldsymbol{x}_i) = \sum_{d=1}^{D} S_d \cdot w_d(\boldsymbol{x}, \boldsymbol{x}_i), \tag{3.4}$$

where the eigengap S_d indicates the importance of each intrinsic dimensionality. Since $S_d \geq 0$ and $\sum_{d=1}^{D} S_d = 1$, we may interpret S_d as the "probability" of the manifold being d-dimensional and the weight value w as the expected weight with respect to this distribution over manifold dimensionalities.

4 Ant Colony: Nonhomogeneous Markov Chain

The group behavior of decentralized natural systems, such as ant colony and bird flocking, has inspired many methods in computer science and is often summarized under the key term *swarm intelligence* (SI). For instance, the biological behavior of ants in the food-seeking process has motivated several methods in combinatorial optimization and clustering. In the process, two mechanisms, one behavioral and one environmental, help to find the shortest path to a food source:

- Deposition: When a bio-ant walks to (and from) a food source, it releases a substance called pheromone on the ground. Ants are attracted by it when they choose a path to follow. The more ants use a route, the more pheromone accumulates, increasing the chance for the path to be selected by subsequent ants as well.
- Evaporation: The pheromone evaporates over time, and hence, less attractive trails with less pheromone eventually disappear.

Thus, although there is no central authority to control the ants' behavior, their indirect form of local interactions, via pheromone, helps them find the shortest path to the closest food source.

In this article, we propose a new algorithm that aims to recover manifolds from noisy samples. The basic idea is that a set of ant-like agents is released in the data space to search for manifolds based on the following elements:

O1: An MC that highlights underlying manifolds in the data set
O2: Deposition and evaporation mechanisms to update the pheromone

The aim is that ants are more likely to visit points close to manifolds and deposit pheromone. Higher pheromone levels function as a positive feedback mechanism, reinforcing further visitations. Eventually, by extracting points with more pheromone, potential manifolds can be uncovered. In the following, we formally define and illustrate the two elements outlined above.

4.1 Manifold Alignment Aware Ants (M3A). Inspired by swarm intelligence, M3A uses several ants that walk in the data space and search for manifolds. During the search process, the ants interact with one another through pheromone and with the environment. In section 3, we proposed possible formulations of transition probabilities for the random walk. However, they consider only the distribution of data points in the environment and do not provide any form of communication among ants. Thus, we modify the transition probabilities by adding a pheromone factor f. Given any data point $\boldsymbol{x}_i$, let f_i denote the amount of pheromone on it, and $\mathcal{N}_i$ represents the set of its neighbors. The new transition probability is defined as

$$p_{ij} = \frac{\left(w_{ij}\right)^{1-\gamma}\left(\hat{f}_j\right)^{\gamma}}{\sum_{\boldsymbol{x}_k \in \mathcal{N}_i} \left(w_{ik}\right)^{1-\gamma}\left(\hat{f}_k\right)^{\gamma}}, \tag{4.1}$$

where $w_{ij} = w(\boldsymbol{x}_i, \boldsymbol{x}_j)$. To ensure that the weights w and the pheromone f are in the same scale, $[0, 1]$, the pheromone is normalized within the neighborhood by $\hat{f}_j = \frac{f_j}{\sum_{\boldsymbol{x}_k \in \mathcal{N}_i} f_k}$. The $\gamma \in [0, 1]$ effectively controls how much ants are attracted to structures already found or are able to explore the data space for new regions. For instance, if we set γ to zero, there would be no interaction among ants, and as a result, every ant could independently explore the data space. Hence, the new transition probability takes into account both types of interactions O1 and O2.

Pheromone is an indirect form of communication among artificial ants that release it on the visited points. Therefore, the amount of pheromone on a sample varies, and it should be updated over time. In order to make the algorithm suitable for parallelization, the pheromone values are updated when the ants finish their walks of a predefined number of steps n, called *one round*.[3] Thus, the transition probability for the $(t+1)$th round can be rewritten as

$$p_{ij}^{(t+1)} = \frac{\left(w_{ij}\right)^{1-\gamma}\left(\hat{f}_j^{(t)}\right)^{\gamma}}{\sum_{k\in\mathcal{N}_i}\left(w_{ik}\right)^{1-\gamma}\left(\hat{f}_k^{(t)}\right)^{\gamma}}, \tag{4.2}$$

where $f_j^{(t)}$ denotes the amount of pheromone on $\boldsymbol{x}_j$ after t rounds. Since the amount of pheromone on a data point depends on the number of visits and the extent of evaporation, the updating rule consists of two parts:

- Deposition based on the number of times a point has been visited
- Evaporation depends on the environment and is controlled by hyperparameter ρ

Therefore, the pheromone level on any sample $\boldsymbol{x}_i$ can be updated as

$$f_i^{(t+1)} = \frac{c}{M}\sum_{a=1}^{M}\frac{N_i^a(n)}{n+1} + (1-\rho)\cdot f_i^{(t)}, \tag{4.3}$$

where M is the number of ants and $N_i^a(n)$ is the number of times the ath ant visits $\boldsymbol{x}_i$ over $n+1$ steps (including initial state) in the $(t+1)$th round. Note that the deposition term is divided by $M(n+1)$ to prevent unlimited increase of the pheromone level (especially for big M or n). The constant $c > 0$ specifies the amount of pheromone a single ant deposits on a point in a single visit. The pseudocode of the M3A[4] is provided in algorithm 2. In summary, we can highlight the following points:

- The proposed ant algorithm uses transition probabilities (see equation 4.2) based on local tangent alignment and pheromone to highlight manifold structures, with the latter also reinforcing the agents to stay close.
- In contrast to previous works with the ant colony, the pheromone is deposited on the nodes instead of edges. This vastly reduces the number of pheromone values that need to be stored. Moreover, the pheromone associated with each data point can be used to extract points close to manifolds.

[3]In one round, we can distribute the agents among several processors and update the pheromone after n steps. Otherwise, communications between processors would be necessary to synchronize pheromone values in every step, which would cause unnecessary overhead.

[4]M3A implementation is provided at https://github.com/mohammadimathstar/M3A.

Algorithm 1: OneAnt.

```
1 Input: pheromone $\mathbf{f} = [f_1, f_2, ..., f_N]$;
2 $\mathbf{N}_v = [0, 0, ..., 0]$ ;
3 Randomly select a node as the initial position (say $\boldsymbol{x}_i$) ;
4 $N_v[i] = N_v[i] + 1$;
5 for $s = 1$ to $n$ do
6     randomly select its next destination following equation 4.1(say $\boldsymbol{x}_j$);
7     $N_v[j] = N_v[j] + 1$;
8 end
  Result: number of visits $\mathbf{N}_v$.
```

- The amount of pheromone is updated after a predefined number of steps n. In practice, this allows distributing the computations among several processors and parallelize the random walks.

4.2 Complexity Analysis. As a preprocessing step for the M3A algorithm, we implement two operations on each data point: (1) finding its neighbors and (2) performing PCA on its neighborhood. To perform the neighbor search we need to calculate the distances of all pairs of N samples, which leads to the complexity of $\mathcal{O}(N^2)$. However, there are approximate nearest neighbor search strategies to reduce the complexity. For instance, the k-d tree algorithm (Bentley, 1975) with the complexity $\mathcal{O}(N \log N)$ can significantly decrease the computational costs. Besides the neighbor search, we perform local PCA using singular value decomposition. In the worst case, its implementation for any point $\boldsymbol{x}$ scales cubic with the dimensionality of the data and quadratic with the size of its neighborhood $|\mathcal{N}_{\boldsymbol{x}}|$. Nonetheless, approximate strategies decrease the computational complexity (Golub & Van Loan, 2012). In addition to speeding up the preprocessing step via approximate strategies, the execution of the M3A can be accelerated via parallelization. We only need to distribute the ants (i.e., function OneAnt introduced in algorithm 1) among several processors.

5 Theoretical Analysis

Many metaheuristic methods suffer from the lack of theoretical analysis. Therefore, their behavior and the effects of parameters on them are typically

Algorithm 2: Manifold Alignment Aware Ants (M3A).

```
1 Initialize the pheromone vector $\mathbf{f} = [f_1^{(0)}, f_2^{(0)}, ..., f_N^{(0)}]$;
2 for $t = 1$ to $N_{iter}$ do
3     $\mathbf{N}_v = [0, 0, ..., 0]$;
4     for $a = 1$ to $M$ do
5         $\mathbf{N}_v^a$ = OneAnt( $\mathbf{f}$ );[OneAnt is introduced in algorithm 1].
6     end
7     $\mathbf{N}_v = \sum_{a=1}^{M} \mathbf{N}_v^a$;
8     Update pheromone by equation 4.3.
9 end
  Result: The pheromone vector $\mathbf{f} = [f_1, f_2, ..., f_N]$.
```

investigated only empirically. In this section, we provide an analysis of the convergence and the impact of hyperparameters of the proposed algorithm in exemplary situations. More precisely, we consider a data set containing a noisy d-dimensional manifold and we use equation 3.3 to compute the weight values with α determined such that $p = 100\%$ of neighbors having nonzero weight values. First, we study the effect of noise on the performance of PCA and, as a result, on the M3A algorithm. Then we examine the pheromone distribution and demonstrate its convergence to the stationary distribution and its capability in recovering a linear manifold. Finally, we study the performance of the algorithm on nonlinear manifolds under the assumption that they can be approximated by local linear patches. We concentrate our analysis on a single patch and consider the other patches as conceptually grouped in a single state representing "the outside."

5.1 Spectral Analysis. An important subject in perturbation theory is to study the effect of noise on the eigenvalues and eigenvectors of matrices (Kaslovsky & Meyer, 2014). Since we use principal directions of covariance matrices to define the MCs, it is important to know the effect of noise on the eigenvectors of covariance matrices. Let $\mathcal{M}$ be a d-dimensional vector subspace embedded in a higher-dimensional space.[5] In the case of

[5]The assumptions are made since we only focus on linear manifolds in the next subsection. For a more general case, nonlinear manifold, we refer to Kaslovsky and Meyer (2014) where the effect of curvature on the covariance matrices is studied.

noise-free samples, it can be exactly recovered via PCA. However, in the presence of noise, the recovered subspace $\hat{\mathcal{M}}$ is perturbed. Here, the goal is to see the impact of noise on $\hat{\mathcal{M}}$.

Without loss of generality, let us consider the first d coordinates that span the subspace $\mathcal{M}$. We assume the set $\{\boldsymbol{l}_i\}_{i=1}^{N}$ contains N realizations of $\mathcal{M}$, such that their first d coordinates $(l_i^{(1)}, \ldots, l_i^{(d)})$ are uniformly distributed within $B_r(\mathbf{0})$. Thus, every point $\boldsymbol{l}_i$ on the subspace $\mathcal{M}$ has the form $(l_i^{(1)}, \ldots, l_i^{(d)}, 0, \ldots, 0)$. Now, suppose these realizations are disrupted by gaussian noise with mean zero and standard deviation σI_D. If $\boldsymbol{x}_i$ denotes a noisy observation, then we have

$$\boldsymbol{x}_i = \boldsymbol{l}_i + \boldsymbol{e}_i,$$

where $\boldsymbol{e}_i$ is the noise vector. In this setting, the design matrix X can be described as

$$X = L + E, \tag{5.1}$$

where the columns of L and E keep the noise-free realizations and noise vectors, respectively. Let us denote the centered version of a matrix $H = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_N]$ as

$$\tilde{H} = H - \mu_h \mathbf{1}_N,$$

where $\mu_h = \frac{1}{N}\sum_i \boldsymbol{h}_i$ and $\mathbf{1}_N = \left[1, 1, \ldots, 1\right]$ with N entries. Then the data covariance matrix can be written as

$$\frac{1}{N}\tilde{X}\tilde{X}^T = \frac{1}{N}\tilde{L}\tilde{L}^T + \Delta,$$

with the matrix Δ representing the perturbation caused by the noise:

$$\Delta = \frac{1}{N}(\tilde{L}\tilde{E}^T + \tilde{E}\tilde{L}^T + \tilde{E}\tilde{E}^T).$$

From eigendecomposition, we obtain

$$\frac{1}{N}\tilde{L}\tilde{L}^T = U\Lambda U^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_1 & U_2 \end{bmatrix}^T,$$

where Λ_1 is a $d \times d$ diagonal matrix containing nonzero eigenvalues in descending order and U_1 includes their corresponding eigenvectors. Note that

U_2 can be any orthogonal basis for the last $D-d$ coordinates. Similarly, we can write

$$\frac{1}{N}\tilde{X}\tilde{X}^T = \hat{U}\Lambda\hat{U}^T = \begin{bmatrix}\hat{U}_1 & \hat{U}_2\end{bmatrix}\begin{bmatrix}\hat{\Lambda}_1 & 0\\ 0 & \hat{\Lambda}_2\end{bmatrix}\begin{bmatrix}\hat{U}_1 & \hat{U}_2\end{bmatrix}^T.$$

The subspace $\mathcal{M}$ is spanned by U_1, and the recovered subspace $\hat{\mathcal{M}}$ is spanned by $\hat{U}_1$. The orthogonal projectors onto $\mathcal{M}$ and $\hat{\mathcal{M}}$ are derived as follows:

$$Q = U_1U_1^T \qquad \hat{Q} = \hat{U}_1\hat{U}_1^T.$$

Here, the Frobenius distance $\|Q-\hat{Q}\|_F$ is used to compare two subspaces because it corresponds to the sum of the squared sines of the principal angles between $\mathcal{M}$ and $\hat{\mathcal{M}}$ (Kaslovsky & Meyer, 2014). If the number of samples N and the probability constants ε and ε_λ satisfy the following inequalities,

$$N > 4\left(\max(\sqrt{d}, \sqrt{D-d}) + \varepsilon\right), \quad \varepsilon < 0.7\sqrt{d(D-d)}, \quad \varepsilon_\lambda < \frac{3}{\sqrt{d+2}}\sqrt{N} \tag{5.2}$$

then the following theorem offers a bound on the angle between $\mathcal{M}$ and $\hat{\mathcal{M}}$.

Theorem 4 *(Kaslovsky & Meyer, 2014). Let*

$$\delta = \frac{r^2}{d+2}\left(1 - \frac{1}{\sqrt{N}}\zeta_1(\varepsilon_\lambda)\right) - \sigma\frac{1}{\sqrt{N}}\zeta_2(\varepsilon_\lambda) - \sigma^2\left(\sqrt{d} + \sqrt{D-d} + \frac{1}{\sqrt{N}}\zeta_3(\varepsilon)\right)$$

and

$$\beta = \frac{1}{\sqrt{N}}\left[\sigma\sqrt{d(D-d)}\eta(\varepsilon,\varepsilon_\lambda) + \frac{1}{\sqrt{N}}\zeta_{\text{numer}}(\varepsilon)\right].$$

Additionally, if the following conditions hold:

- *(Condition 1)* $\delta > 0$,
- *(Condition 2)* $\beta < \frac{1}{2}\delta$

then

$$\|Q - \hat{Q}\| \le \frac{2\sqrt{2}\beta}{\delta} \tag{5.3}$$

with probability greater than

$$1 - 2de^{-\varepsilon_\lambda^2} - 9e^{-\varepsilon^2} \tag{5.4}$$

over the joint random selection of the sample points and random realization of the noise, where the following definitions have been made to ease the presentation:

$$\begin{aligned}
\zeta_1(\varepsilon_\lambda) &= \frac{2}{\sqrt{N}} - \frac{1}{N^{\frac{3}{2}}} + \left(1 - \frac{1}{N}\right)\varepsilon_\lambda\sqrt{8(d+2)},\\
\zeta_2(\varepsilon, \varepsilon_\lambda) &= \frac{2rd}{\sqrt{d+2}}\left(1 + \varepsilon_\lambda\frac{5\sqrt{d+2}}{\sqrt{N}}\right)\left(1 + \frac{6\varepsilon}{5d}\right),\\
\zeta_3(\varepsilon) &= \frac{5}{2}\left(\sqrt{d} + \varepsilon\sqrt{2}\right)\left(\sqrt{D-d} + \varepsilon\sqrt{2}\right),\\
\eta(\varepsilon, \varepsilon_\lambda) &= \left(1 + \frac{6}{5}\frac{\varepsilon}{\sqrt{d(D-d)}}\right)\left[\sigma + \frac{r}{\sqrt{d+2}}\left(1 + \varepsilon_\lambda\frac{5\sqrt{d+2}}{\sqrt{N}}\right)\right],\\
\zeta_{\text{numer}}(\varepsilon) &= \sigma^2\sqrt{d(D-2)}\left(1 + \frac{6}{5}\frac{\varepsilon}{\sqrt{d(D-d)}}\right)\left(\sqrt{D-d} + \varepsilon\sqrt{2}\right).
\end{aligned}$$

This theorem provides an upper bound for the difference between the true manifold $\mathcal{M}$ and the recovered subspace $\hat{\mathcal{M}}$. Therefore, it is safe to say that the definition of weights, in equation 3.3, is reasonable in keeping a random walker close to the $\mathcal{M}$ (by encourage to stay close to $\hat{\mathcal{M}}$). Moreover, theorem 4 can help us to explain the relation between the quality of the recovered subspace $\hat{\mathcal{M}}$ and the hyperparameters N, σ and r. An empirical example demonstrates the relationship in practice. We generate samples from a noisy one-dimensional linear manifold embedded in $\mathbb{R}^{10}$ for various (hyper-) parameter settings. Their influence is demonstrated in Figure 2. Figure 2a shows that the approximation performance increases with a growing number of samples N. On the other hand, there is a direct connection between the noise level σ and the approximation error as shown in Figure 2b. Moreover, Figure 2c demonstrates that a bigger neighborhood radius r recovers an increasingly accurate subspace, which is not the case for nonlinear manifolds, where this also depends on the curvature.

5.2 Pheromone Distribution for a Linear Manifold. The proposed M3A algorithm associates a pheromone value with each sample. In this section, we show that the pheromone values encode distances of the samples to the reconstructed subspace $\hat{\mathcal{M}}$. Let the data set fulfill the conditions just explained—that the noisy data points are uniformly distributed along a linear manifold inside $B_r(\mathbf{0})$. In order to simplify our analysis, we make the following assumptions in constructing the MC:

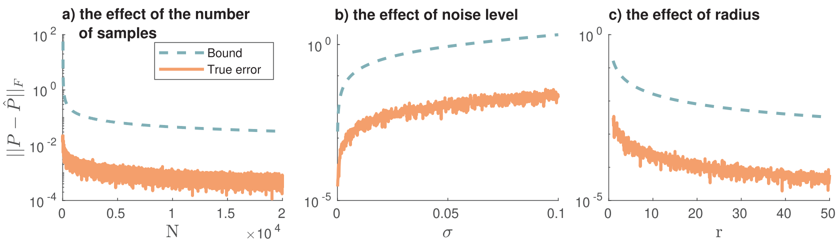

Figure 2: The effect of (hyper-)parameters on the quality of the recovered subspaces: (a) the effect of the number of samples N within the neighborhood ($\sigma = 0.01$, $r = 1$), (b) the role of the noise level σ ($N = 1000$, $r = 1$), and (c) the impact of neighborhood radius r ($N = 1000$, $\sigma = 0.01$).

- The radius r is big enough such that all pairs of samples are neighbors and any sample belongs to its neighborhood. Therefore, instead of using local PCA for each point, we apply PCA on the whole data set.
- In equation 3.2 we set $\boldsymbol{x}'$, to the origin $\mathbf{0}$. Thus, for any point $\boldsymbol{x}_i$, its distance to $\hat{\mathcal{M}}$ is

$$\delta_i^{\mathcal{M}} = \|(I - \hat{U}_1\hat{U}_1^T)\boldsymbol{x}_i\|,$$

and with equation 3.3, the weight values are computed as

$$w_{ki} = 1 - \frac{\delta_i^{\mathcal{M}}}{\alpha} = w_i \qquad 1 \le k \le N. \tag{5.5}$$

We define $w_i = w_{ki}$ since w_{ki} does not depend on k.

- Since the presence of powers in the transition probabilities (i.e., γ and $1-\gamma$) makes our analysis intractable, we analyze the special case with removed powers, such that our arguments can be presented in a tractable manner:

$$p_{ji}^{(t+1)} = \frac{w_i f_i^{(t)}}{\sum_k w_k f_k^{(t)}}. \tag{5.6}$$

- The pheromone values are uniformly initialized by

$$f_i^{(0)} = \frac{1}{N} \qquad \forall i. \tag{5.7}$$

- In the updating rule of pheromone, equation 4.3, we set the constant c to the evaporation rate ρ:

$$f_i^{(t+1)} = \frac{\rho}{M}\sum_{a=1}^{M}\frac{N_i^a(n)}{n+1} + (1-\rho)f_i^{(t)}. \tag{5.8}$$

Without loss of generality we re-label the data points by $\{1, \ldots, N\}$ such that

$$w_1 \geq w_2 \geq \cdots \geq w_N. \tag{5.9}$$

Thus, the new labels sort samples according to their closeness to the subspace $\hat{\mathcal{M}}$.

Corollary 1. *The pheromone values across the samples form a probability distribution called pheromone distribution denoted by* $\mathbf{f}^{(t)} = \left[f_1^{(t)}, f_2^{(t)}, \ldots, f_N^{(t)} \right]$.

Proof. It can be proven by induction over t. From equation 5.7, we have $\sum_{i=1}^{N} f_i^{(0)} = 1$, and if we assume that the statement is valid for t, that is, $\sum_{i=1}^{N} f_i^{(t)} = 1$, we can rewrite equation 5.8:

$$\sum_{i=1}^{N} f_i^{(t+1)} = \frac{\rho}{M} \sum_{a=1}^{M} \left(\overbrace{\frac{1}{n+1} \sum_{i=1}^{N} N_i^a(n)}^{=1} \right) + 1 - \rho = 1.$$

Note that the expression inside the parentheses is equal to 1 since $\sum_{i=1}^{N} N_i^a(n)$ equals to the number of steps (i.e., n) plus 1 (for the initial point). □

In this setting we show that:

- The pheromone values on sample points are sorted by their distances to the subspace $\hat{\mathcal{M}}$.
- The pheromone distribution $\mathbf{f}^{(t)}$ converges as $t \to \infty$.

Our analysis is based on the Markov chain framework, and we consider two cases. First, we assume there is only one ant that walks for an unbounded number of steps in every round (i.e., $n \to \infty$). Second, we assume an unbounded number of ants (i.e., $M \to \infty$) that walk for n steps each in every round. In other words, we study the long-term and the short-term behavior of random walks, respectively.

5.2.1 Single Ant, Unbounded Path Length. We assume a single ant (i.e., $M = 1$) performing $n \to \infty$ steps in each round. From theorem 3, the long-term fraction of time spending in a point $\boldsymbol{x}_i$ is equal to its stationary distribution value π_i. Therefore, for any point $\boldsymbol{x}_i$, its pheromone value is updated according to

$$\begin{aligned} f_i^{(t+1)} &= \frac{\rho}{M} \sum_{a=1}^{M} \pi_i^{(t+1)} + (1-\rho) f_i^{(t)} \\ &= (1-\rho) f_i^{(t)} + \rho \cdot \pi_i^{(t+1)}, \end{aligned} \tag{5.10}$$

where $\pi^{(t+1)}$ is the stationary distribution associated to the transition probability matrix in the $(t+1)$th round. The stationary distribution can be computed via the balance equation:

$$\begin{aligned}\pi_i^{(t+1)} &= \sum_j \pi_j^{(t+1)} p_{ji}^{(t+1)} \\ &= \sum_j \pi_j^{(t+1)} \frac{w_i f_i^{(t)}}{\sum_k w_k f_k^{(t)}} \\ &= \frac{w_i f_i^{(t)}}{\sum_k w_k f_k^{(t)}} \\ &= p_{ji}^{(t+1)} \qquad \forall j. \end{aligned} \tag{5.11}$$

From equations 5.7 and 5.9, it can be shown (by induction) for any $t > 0$:

$$\pi_1^{(t)} \geq \pi_2^{(t)} \geq \cdots \geq \pi_N^{(t)} \tag{5.12}$$

and by equation 5.10,

$$f_1^{(t)} \geq f_2^{(t)} \geq \cdots \geq f_N^{(t)}. \tag{5.13}$$

Since the weight values encode the distance of samples to the subspace $\hat{\mathcal{M}}$ (see equation 3.3), the following corollary holds.

Corollary 2. *The pheromone values are sorted according to their closeness to the linear manifold* $\hat{\mathcal{M}}$.

In the next step, we would like to prove the convergence of the pheromone distribution, but we need to first establish the following lemma:

Lemma 1. *The sequence* $\{a^{(t)}\}_{t=0}^{\infty}$ *with*

$$a^{(t)} = \sum_k w_k f_k^{(t)} \tag{5.14}$$

- *Is monotonically increasing (i.e.,* $a^{(t+1)} \geq a^{(t)}$*)*
- *Is convergent (i.e.,* $a^{(t)} \to a$ *for some* $a \geq 0$*)*

Proof. (a) To demonstrate that the sequence $\{a^{(t)}\}_{t=0}^{\infty}$ is monotonically increasing, we need to show

$$I^{(t)} = a^{(t+1)} - a^{(t)} = \sum_k w_k \cdot \Delta^{(t)} f_k \geq 0,$$

where $\Delta^{(t)} f_i = f_i^{(t+1)} - f_i^{(t)}$. We can write

$$\begin{aligned}
\Delta^{(t)} f_i &\overset{\text{Eq. (25)}}{=} \rho(\pi_i^{(t+1)} - f_i^{(t)}) \\
&\overset{\text{Eq. (26)}}{=} \rho\left(\frac{w_i}{\sum_k w_k f_k^{(t)}} - 1\right) f_i^{(t)} \\
&= \frac{\rho}{a^{(t)}}\left(w_i - a^{(t)}\right) f_i^{(t)}. \qquad (5.15)
\end{aligned}$$

For every fixed t, we have: $\{f_i^{(t)}\}_{i=1}^N$ and $\{w_i - a^{(t)}\}_{i=1}^N$[6] that are monotonically decreasing sequences. Thus, the sequence $\{\Delta^{(t)} f_i\}_{i=1}^N$ will be decreasing as well. Therefore, we have

$$I^{(t)} = \sum_{k \le k^*} w_k \cdot \overbrace{\Delta^{(t)} f_k}^{\ge 0} + \sum_{k > k^*} w_k \cdot \overbrace{\Delta^{(t)} f_k}^{<0},$$

where $k^* = \max\{k : \Delta^{(t)} f_k \ge 0\}$, and we define the following values:

$$w = \min_{k \le k^*} w_k, \quad w' = \max_{k > k^*} w_k.$$

Since $\{w_i\}_{i=1}^N$ is decreasing, we have $w > w'$:

$$\begin{aligned}
I^{(t)} &\ge w \cdot \sum_{k \le k^*} \overbrace{\Delta^{(t)} f_k}^{\ge 0} + w' \cdot \sum_{k > k^*} \overbrace{\Delta^{(t)} f_k}^{<0} \\
&\overset{w > w'}{>} w \cdot \sum_{k \le k^*} \Delta^{(t)} f_k + w \cdot \sum_{k > k^*} \Delta^{(t)} f_k \\
&= w \cdot \sum_k \Delta^{(t)} f_k = w \cdot \sum_k (f_k^{(t+1)} - f_k^{(t)}) \\
&= w \cdot \left(\sum_k f_k^{(t+1)} - \sum_k f_k^{(t)}\right) = 0.
\end{aligned}$$

(b) We know that

$$0 \le w_i \le 1, \qquad 0 \le f_i^{(t)} \le 1$$

[6]From inequality 5.9, we know that $\{w_i\}_{i=1}^N$ is a decreasing sequence. Therefore, the sequence $\{w_i - a^{(t)}\}_{i=1}^N$ is also decreasing, since $a^{(t)}$ is a constant number for a fixed t.

and therefore

$$0 \leq a^{(t)} \leq N.$$

Since $a^{(t)}$ is an increasing and bounded sequence, it is convergent. □

The following theorem states that the pheromone values are convergent over time, and, as a result, the algorithm is convergent to a unique distribution.

Theorem 5. *For any point $\mathbf{x}_i$, its pheromone value $f_i^{(t)}$ is convergent as $t \rightarrow \infty$.*

Proof. Since we know that the pheromone values $f_i^{(t)}$ are bounded from below and above for all t, it is enough to show that $\{f_i^{(t)}\}_{t=1}^{\infty}$ is a monotonic sequence. From equation 5.15, we just need to show that $w_i - a^{(t)}$ becomes only positive or only negative after some round T. Let us define the sequence $\{b^{(t)}\}$ as $b^{(t)} = w_i - a^{(t)}$. From lemma 1 we know sequence $a^{(t)}$ is increasing and $a^{(t)} \rightarrow a$ (as $t \rightarrow \infty$). Thus, $b^{(t)}$ will be a decreasing sequence and convergent to $w_i - a$:

- If $w_i \geq a$, then $b^{(t)} \geq 0$ for any t. Thus, from equation 5.15, we get $\Delta^{(t)} f_i \geq 0$ and then $f_i^{(t)}$ is a monotonically increasing sequence.
- If $w_i < a$, then there exists $T \in \mathbb{N}$ such that

$$\forall t > T : a^{(t)} > w_i \Longrightarrow b^{(t)} < 0,$$

so $f_i^{(t)}$ is a decreasing sequence for $t > T$. □

Corollary 3. *From the convergence of pheromone values and the updating rule in equation 5.10 it is clear that $\pi_i^{(t)}$ is convergent (as $t \rightarrow \infty$) for any $i \in \{1, 2, \ldots, N\}$.*

After proving the convergence of the pheromone distribution and stationary distribution, we show in the following that the pheromone distribution (coming from the algorithm) converges to the stationary distribution.

Theorem 6. *The pheromone distribution $\boldsymbol{f}^{(t)}$ converges to the stationary distribution $\pi^{(t)}$ as $t \rightarrow \infty$, that is, for any $\epsilon > 0$, there exists $T \in \mathbb{N}$ such that*

$$\forall i \in \{1, \ldots, N\},\ \forall t > T \quad \rightarrow \quad |\pi_i^{(t)} - f_i^{(t)}| < \epsilon.$$

Proof. Theorem 5 states that the pheromone distribution converges (i.e., $\forall i \ \ f_i^{(t)} \rightarrow f_i$ as $t \rightarrow \infty$). Therefore, the update rule can be written as

$$f_i = (1 - \rho) f_i + \rho \cdot \lim_{t \rightarrow \infty} \pi_i^{(t)} \Longrightarrow \lim_{t \rightarrow \infty} \pi_i^{(t)} = f_i.$$

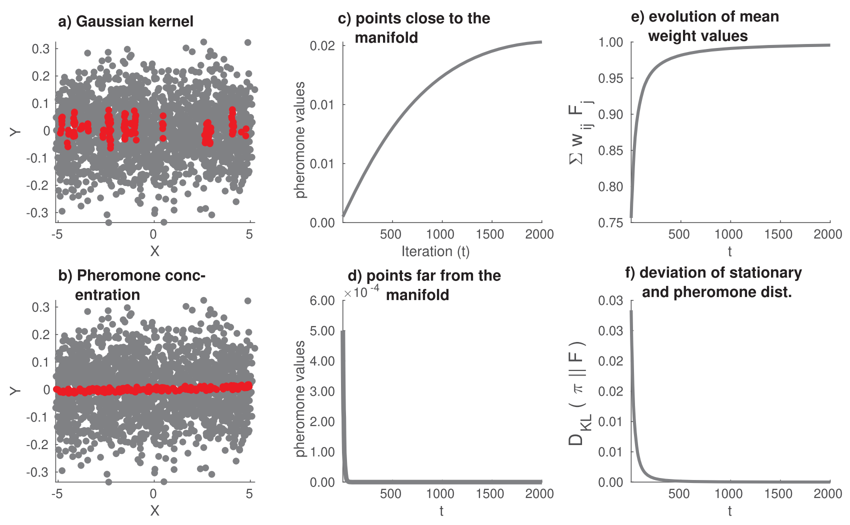

Figure 3: 1D manifold consisting of 2000 data points disrupted by gaussian noise ($N(0, 0.1)$) depicted as gray points in panels a and b. The 120 points with highest density (using a gaussian kernel) and pheromone values (resulting from equation 5.5) are highlighted in red in panels a and b, respectively. The development of the pheromone values over iterations t for points close to the manifold is steeply increasing (shown in panel c), while it is abruptly dropping to zero for points far away (see panel d). Panel e shows the evolution of the mean value of weights $a^{(t)}$ (see equation 5.14) and f the KL-divergence from the stationary to the pheromone distribution over the iterations t.

Thus, the pheromone and stationary distribution converge toward the same distribution. $\square$

Theorems 5 and 6 indicate two important points: that the pheromone distribution is convergent and that $f^{(t)}$ converges to a well-known distribution called the stationary distribution.

In order to demonstrate the above results with an example, we generate a 1D manifold such that 2000 data points are uniformly distributed on a line and disrupted by gaussian noise $N(0, 0.1)$[7] (gray points as shown in Figures 3a and 3b). In Figure 3a, the gaussian kernel with $\sigma = 1$ and

[7] Other noise models, such as uniform noise ("tubes" around the manifold) or the Laplace distribution (strongly concentrating points along the manifold) can be alternative scenarios dependent on the process with which the manifolds are sampled. However, here we concentrate on gaussian noise that is suitable for our applications to keep the article more concise.

$r = 0.1$[8] is used to detect 120 samples with the highest density highlighted in red. Figure 3b shows the samples with the highest pheromone values using the weights defined in equation 5.5. With the gaussian kernel failing, we clearly see the superiority of the new algorithm in recovering the linear manifold nearly perfectly, since it uses PCA in the creation of the MC. Figures 3c and 3d depict the evolution of pheromone values of samples close to and far from the underlying 1D manifold, respectively. Closer samples to the manifold exhibiting lower noise levels receive more and more visits in smaller time intervals (from theorem 3) and thus accumulate more and more pheromone over time. Figure 3e displays that the mean of weight values $a^{(t)} = \sum_k w_k f_k$ is monotonic increasing and convergent, as explained in lemma 1. Finally, Figure 3f uses the Kullback-Leibler (KL) divergence to show that the pheromone distribution converges toward the stationary distribution, confirming theorem 6.

5.2.2 Unbounded Number of Ants, Fixed Bounded Path Length. Since our algorithm allows employing multiple ants, we study its performance in the presence of a swarm of ants, which walk for a limited number of steps n in every round. Let M denote the number of ants and $N_j(n)$ represents the number of times a random walker visits point $\boldsymbol{x}_j$ during n steps on $(t+1)$th round. For a specific case $M = 1$, we have (according to definition 1),

$$E[N_j(n)] = E_{P(X_0=\boldsymbol{x}_i)}[m_{ij}(n)], \tag{5.16}$$

and assuming uniform distribution over the initial states (sample points),

$$E[N_j(n)] = \frac{1}{N}\sum_{i=1}^{N} m_{ij}(n). \tag{5.17}$$

For $M > 1$, we still have the above equality for each ant since pheromone values are kept fixed over one round. Hence, in each round, an ant walks independently and others do not have any impact on it. Initializing each ant on one of the sample points with probability $P(X_0 = \boldsymbol{x}_i)$, we obtain by the law of large numbers,

$$\frac{\sum_{a=1}^{M} N_j^a(n)}{M} \to E[N_j(n)], \qquad \text{as } M \to \infty. \tag{5.18}$$

[8] Note that we select r such that the red points are more distributed along the manifold. For bigger value for r, the red points are more concentrated around $x = 0$ and fail to recover the 1D manifold.

As the number of ants M goes to infinity and from equations 5.17 and 5.18, the updating rule, in equation 5.8, can be rewritten as

$$f_j^{(t+1)} = (1-\rho)f_j^{(t)} + \frac{\rho}{N}\sum_{i=1}^{N}\frac{m_{ij}(n)}{n+1}. \tag{5.19}$$

The following corollary says that, like the weight values, the pheromone values form a decreasing sequence.

Corollary 4. *For each t, the sequence of pheromone values $\{f_i^{(t)}\}_{i=1}^{N}$ is monotonically decreasing.*

Proof. We use induction on t to show that the following statement holds for any arbitrary pair of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ with $i < j$:

$$f_i^{(t)} \geq f_j^{(t)}. \tag{5.20}$$

For simplicity, we drop the exponent t in the following computations. In the first round, $t = 1$, we have

$$f_i^{(0)} = f_j^{(0)} = \frac{1}{N} \overset{w_i \geq w_j}{\Longrightarrow} p_{ki} \geq p_{kj} \qquad \forall k$$

and

$$p_{ki}^{(2)} = \sum_l p_{kl}p_{li} \geq \sum_l p_{kl}p_{lj} = p_{kj}^{(2)},$$

where $p_{ki}^{(2)}$ is the (k, i)th element of the second power of the transition matrix P. Similarly, it can be generalized for any q:

$$p_{ki}^{(q)} \geq p_{kj}^{(q)} \qquad \forall k.$$

As a result of theorem 2, we derive

$$m_{ki}(n) \geq m_{kj}(n) \qquad \forall k.$$

Thus, the updating rule (see equation 5.19) shows that the inequality equation 5.20 holds in the first round. Now let the statement hold up to the tth round (i.e., $f_i^{(t)} \geq f_j^{(t)}$). In the $(t+1)$th round, we have

$$p_{ki} = \frac{w_i f_i}{\sum w_k f_k} \overset{w_i \geq w_j}{\geq} \frac{w_j f_j}{\sum w_k f_k} = p_{kj}.$$

Similar to the first round, we obtain for any q:

$$\forall k \qquad p_{ki}^{(q)} \geq p_{kj}^{(q)} \Longrightarrow m_{ki}(n) \geq m_{kj}(n) \Longrightarrow f_i^{(t+1)} \geq f_j^{(t+1)}.$$

□

Therefore, the pheromone values, similar to weights, encode information about how close the individual sample points are to the subspace $\hat{\mathcal{M}}$. Analogous to theorems 5 and 6, theorem 7 guarantees the convergence of the pheromone and the stationary distributions.

Theorem 7. *The following statements hold for the pheromone distribution:*

(a) *For any point $\mathbf{x}_i$ its pheromone values $f_i^{(t)}$ converge in time t (i.e., $f_i^{(t)} \rightarrow f_i$ as $t \rightarrow \infty$).*
(b) *The pheromone distribution $\boldsymbol{f}^{(t)}$ converges to the stationary distribution $\pi^{(t)}$ as $t \rightarrow \infty$, i.e., for any $\epsilon > 0$ there exists $T \in \mathbb{N}$ such that*

$$\forall i \in \{1, \dots, N\}, \forall t > T \quad \rightarrow \quad |\pi_i^{(t)} - f_i^{(t)}| < \epsilon.$$

Proof. The proof of part a follows that of theorem 5.

(b) From the convergence of pheromone values and the updating rule equation 5.19, we can write:

$$f_i = \frac{1}{N(n+1)} \sum_{i=1}^{N} \lim_{t\rightarrow\infty} m_{ji}^{(t)}(n).$$

From equation 5.11, we know

$$\forall i, j; \qquad p_{ji}^{(t)} = \pi_i^{(t)} \Longrightarrow p_{ji}^{(t)(q)} = \pi_i^{(t)} \qquad \forall q, \tag{5.21}$$

where $p_{ji}^{(t)(q)}$ is the (j, i)th element of $(P^{(t)})^q$ in the tth round. Therefore, we have

$$f_i = \lim_{t\rightarrow\infty} \left(\frac{1}{N(n+1)} \sum_{i=1}^{N} m_{ji}^{(t)}(n) \right) \overset{\text{theo. (2)}}{=} \lim_{t\rightarrow\infty} \left(\frac{1}{N(n+1)} \sum_{i=1}^{N} \sum_{q=0}^{n} p_{ji}^{(t)(q)} \right)$$
$$\overset{\text{Eq. (36)}}{=} \lim_{t\rightarrow\infty} \frac{1}{N(n+1)} \sum_{i=1}^{N} \sum_{q=0}^{n} \pi_i^{(t)} \quad = \lim_{t\rightarrow\infty} \pi_i^{(t)}.$$

□

Hence, it can be said that both scenarios ($n \rightarrow \infty$ or $M \rightarrow \infty$) result in the same pheromone structure and are asymptotic:

- Pheromone values are sorted according to their closeness to the recovered linear manifold.

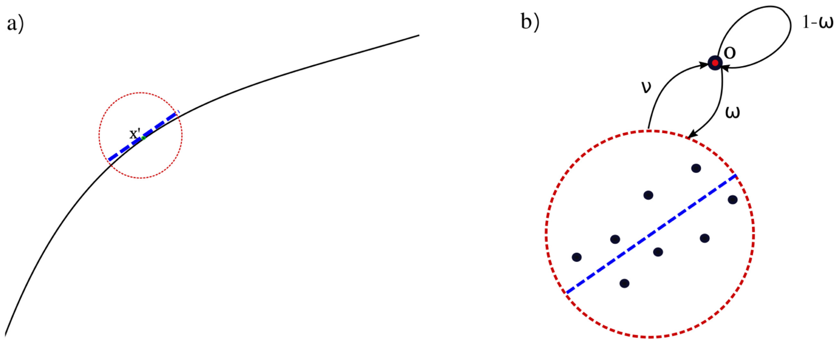

Figure 4: (a) Approximation of a piece of nonlinear manifolds with a linear patch. (b) The linear patch with an external node ($\boldsymbol{o}$), simulating points outside the neighborhood.

- They are convergent to the stationary distribution of the Markov chain.

5.3 Pheromone Distribution for a Nonlinear Manifold. For nonlinear manifolds, we consider a simplified scenario for our analysis. Although it is slightly different from the algorithm's definition, we use it as an example to explain the success of M3A in recovering nonlinear manifolds. Motivated by the assumption that a nonlinear manifold can locally be approximated by linear patches (see Figure 4a), we focus on a small part of the manifold (red circle), and we model all points outside via a single state called $\boldsymbol{o}$. Therefore, we assume that the random walker can jump from any point inside the red circle to $\boldsymbol{o}$ with a probability proportional to the fixed-number ν and vice versa with a probability proportional to the fixed-number ω (see Figure 4b). Consequently, the transition probabilities are defined as

$$p'^{(t+1)}_{\boldsymbol{o}j} = \begin{cases} \frac{\omega}{N}, & \text{if } j \neq \boldsymbol{o} \\ 1-\omega & \text{if } j = \boldsymbol{o} \end{cases} \quad \text{and} \quad p'^{(t+1)}_{i(\neq \boldsymbol{o})j} = \begin{cases} \frac{\nu}{\sum_{k\neq \boldsymbol{o}} w_k f_k^{(t)} + \nu}, & \text{if } j = \boldsymbol{o} \\ \frac{w_j f_j^{(t)}}{\sum_{k\neq \boldsymbol{o}} w_k f_k^{(t)} + \nu}, & \text{if } j \neq \boldsymbol{o} \end{cases},$$

where N is the number of data points inside the red circle. Note that in order to differentiate the transition probabilities and the stationary distribution in this section from the previous one, we use the prime symbol ($'$).

From the balance equation, we can compute the stationary distribution. Thus, for any state j ($\neq \boldsymbol{o}$), we have

$$\pi'^{(t+1)}_j = \sum_{i\neq \boldsymbol{o}} \pi'^{(t+1)}_i p'^{(t+1)}_{ij} + \pi'^{(t+1)}_{\boldsymbol{o}} p'^{(t+1)}_{\boldsymbol{o}j}$$

$$= \sum_{i \neq o} \pi_i'^{(t+1)} \frac{w_j f_j^{(t)}}{\sum_k w_k f_k^{(t)} + \nu} + \pi_o'^{(t+1)} \frac{\omega}{N}$$

$$= \frac{w_j f_j^{(t)}}{\sum_k w_k f_k^{(t)} + \nu} \sum_{i \neq o} \pi_i'^{(t+1)} + \pi_o'^{(t+1)} \frac{\omega}{N}$$

$$= \left(\frac{w_j f_j^{(t)}}{\sum_k w_k f_k^{(t)}} \right) \left(\frac{\sum_k w_k f_k^{(t)}}{\sum_k w_k f_k^{(t)} + \nu} \right) (1 - \pi_o'^{(t+1)}) + \pi_o'^{(t+1)} \frac{\omega}{N}. \quad (5.22)$$

In the previous section, we found $p_{ij}^{(t+1)} = \frac{w_j f_j^{(t)}}{\sum_k w_k f_k^{(t)}}$ and hence can rewrite the terms above as follows:

$$\pi_j'^{(t+1)} = p_{ij}^{(t+1)} \left(\frac{\sum_k w_k f_k^{(t)}}{\sum_k w_k f_k^{(t)} + \nu} \right) (1 - \pi_o'^{(t+1)}) + \pi_o'^{(t+1)} \frac{\omega}{N}$$

$$= a p_{ij}^{(t+1)} + b, \text{ where}$$

$$a = \left(\frac{\sum_k w_k f_k^{(t)}}{\sum_k w_k f_k^{(t)} + \nu} \right) (1 - \pi_o'^{(t+1)}) > 0$$

$$\text{and } b = \pi_o'^{(t+1)} \frac{\omega}{N} > 0. \quad (5.23)$$

From equation 5.11, we obtain

$$\pi_j'^{(t+1)} = a \pi_j^{(t+1)} + b, \quad (5.24)$$

where $\pi_j^{(t+1)}$ denotes the stationary distribution of the random walk in the previous section. Thus, this equation connects the stationary distributions of the new MC π' to π in the previous section. In other words, the new distribution is monotonically increasing with respect to the previous one.

Theorem 3 connects the long-term behavior of a random walk to its stationary distribution. From equations 5.12 and 5.24, we see that the stationary distribution (and similarly the pheromone distribution) is sorted according to the distances of data points (inside the neighborhood) to the linear patch constructed by PCA. From theorem 2, we know that the occupancy time (i.e., the short-term behavior) is directly connected to the transition probabilities of the MC. For any i and j ($\neq \boldsymbol{o}$), we have

$$p_{ij}'^{(t+1)} = \frac{w_j f_j}{\sum_k w_k f_k + \nu} = \left(\frac{w_j f_j}{\sum_k w_k f_k} \right) \left(\frac{\sum_k w_k f_k}{\sum_k w_k f_k + \nu} \right) = a p_{ij}^{(t+1)}.$$

Thus, the new transition probabilities are a monotonically increasing function of the previous transition probabilities. Therefore, similar results as in section 5.2.2 hold.

By simplifying the scenario, we show in this section that the algorithm sorts data points according to their distances to the linear patches (as estimators for a manifold). From Little et al. (2017), we can conclude that the performance of the new algorithm depends on several factors, including selecting an appropriate neighborhood size r. Note that r should be big enough to include enough samples for recovering the tangent space and small enough to prevent covering high curvature. Therefore, in the presence of an appropriate r value, our analysis shows that the pheromone distribution contains valuable information in order to highlight nonlinear manifolds.

6 Experiments

In this section, we use synthetic and real-world data sets to investigate the performance of M3A in different scenarios. First, we discuss the influence and specification of its hyperparameters. Furthermore, complementary to our analysis in section 5.1, we empirically examine the impact of the new MC formulations for denoising and visualization of nonlinear manifolds. Moreover, we show how the proposed algorithm helps to discover manifolds and build better probabilistic models in the sense of sparseness, descriptiveness, and preservation of structural details.

6.1 Strategy for Hyperparameter Selection. Our analysis in section 5 shows that the neighborhood size r, the number of ants M, and the number of steps n in one round play a major role for the performance of M3A. Thus, an automatic strategy to find appropriate values for these quantities is highly desirable. In the following, we present such strategies for practical application.

In section 5.1, we demonstrate that a proper value for r depends on the noise level σ of the manifold. We observe that a bigger r helps to reduce the effect of the noise and achieve a better approximation for the linear manifold. However, as shown in Kaslovsky and Meyer (2014) and Little et al. (2017) for nonlinear manifolds, the presence of curvature encourages using a smaller radius r such that the manifold looks almost linear within the local neighborhood. Hence, the selection of r is a trade-off between the noise level and curvature of the manifolds. Here, to specify a suitable value for r, we use a mixture of gaussian models to assess quantitatively the recovery of the manifold. First, we place the centers of gaussian distributions on points with higher pheromone values, and then their covariance matrices are computed based on the local neighbors. Finally, we use the average log-likelihood (ALL) function to evaluate how good models (for each r) fit the data set and to pick the best r (see Figure 5c).

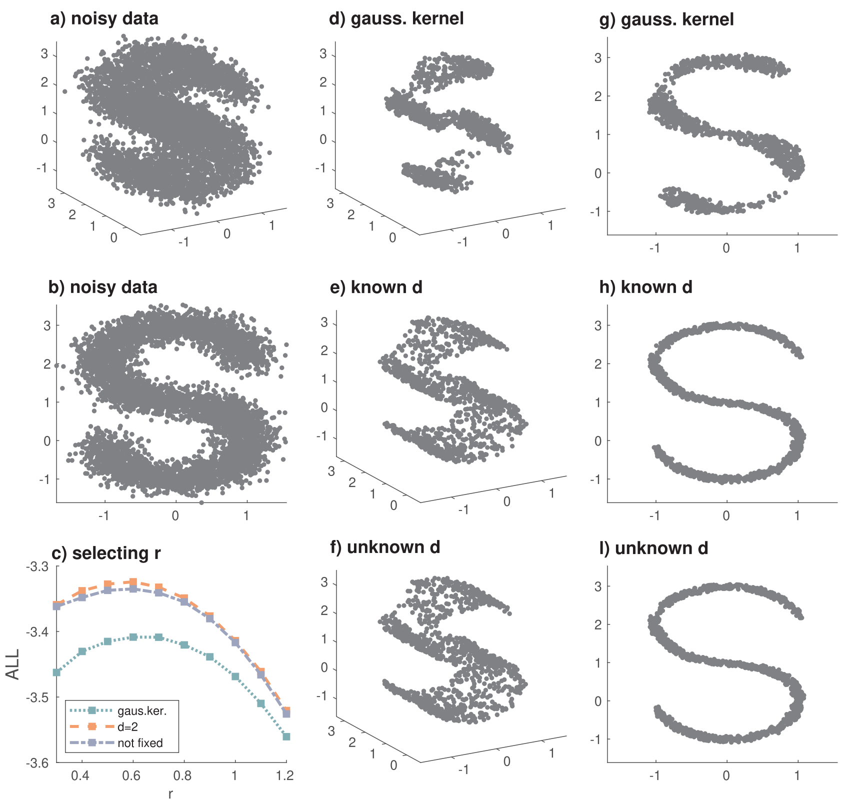

Figure 5: Two views of the noisy s-curve data set (a,b), radius selection via average log likelihood for different r (c), and 1200 points with highest stationary distributions values for the MCs defined by equation 3.3 (e,h) and equation 3.4 (f,l), using $r = 0.6$.

In section 5.2 we investigate two strategies: using a few ants walking many steps in every round, and using many ants walking a limited number of steps. As a rule of thumb, to determine a suitable number of steps and ants (n and M), we assume they fulfill the following inequality,

$$n \cdot M \geq z \cdot N, \tag{6.1}$$

for a constant z. Thus, one may use a small number of workers M (or processors/cores) with a big number of steps n or use many workers with a small number of steps in each round. In our experimentation, we set the number of steps and the number of ants to $n = N$ and $M = z = 50$, respectively.

The pheromone determines how much the ants are reinforced to prefer points frequently visited before. And the evaporation rate ρ and parameter γ control the impact of the pheromone on the random walk process, and thus their determination is connected to M and n. If M and n are both small, it means the algorithm has neither enough ants nor enough time to highlight manifolds; thus, our analysis in section 5.2 is not valid any more. In this case, we suggest using smaller ρ and γ to prevent highlighting background noise. In this contribution, we set them to $\rho = \gamma = 0.1$ for all the experiments.

Furthermore, the maximum amount of pheromone c added on a sample per round is fixed to 2, and the factor α for the weight w_d, in equation 3.3, is determined such that only $p = 50\%$ of neighbors have nonzero values. Finally, for any sample $\boldsymbol{x}_i$, we approximate the point $\boldsymbol{x}'$ that touches the underlying manifold by $\boldsymbol{x}' = \frac{1}{|\mathcal{N}_i|} \sum_{\boldsymbol{x}_k \in \mathcal{N}_i} \boldsymbol{x}_k$ (see equation 3.2).

6.2 Homogeneous Markov Chain. In this section we extend our theoretical analysis with an empirical analysis on nonlinear manifolds. First, we demonstrate the denoising capability of the MCs defined in equations 3.3 and 3.4 on highlighting the underlying manifold. Second, we investigate their impact on the performance of t-distributed stochastic neighbor embedding (t-SNE) (Van der Maaten & Hinton, 2008), a widely used dimension reduction and visualization technique.

6.2.1 Denoising the Manifold. To check the success of the weighting methods (see section 3) in keeping ants close to the manifold, we need to study the short, and long-term behavior of the random walk defined by equation 2.2. According to theorems 2 and 3, the short- and long-term behaviors of MCs are directly connected to the power series of the transition probability and to the stationary distribution, respectively. Here, we use a synthetic data set to investigate the constructed MCs.

Figures 5a and 5b show 6000 samples generated from the s-curve manifold disrupted by gaussian noise $N(0, 0.2)$. In order to select r, we follow the strategy introduced in section 6.1. We apply M3A with different r using 20% of the samples (1200 points) with the highest pheromone values as means for gaussian distributions and their neighbors to determine the covariance matrices. The quality of the models is computed by the average log-likelihood (ALL) function, and the results are shown in Figure 5c. Figures 5d and 5l show 1200 points with the highest stationary distribution values for different MCs. It can be seen that the MC based on the gaussian kernel function fails to recover the manifold (see panels d and g). However, the other methods are successful in keeping random walkers close to the manifold (see panels e, f, h, and l). Note that panels e and f also show that since we use our knowledge about the intrinsic dimensionality of the manifold in equation 3.3 it gives a slightly better result than the MC defined by equation 3.4. A similar result is achieved for the short-term behavior of the MCs where the formulation of equations 3.3 and 3.4 outperforms the gaussian kernel function.

6.2.2 Manifold Aligned Similarities in t-SNE. A specific case of dimensionality reduction is the visualization where high-dimensional data are embedded in two or three dimensions. A well-known nonlinear tool to visualize high-dimensional data is a t-distributed stochastic neighbor embedding (t-SNE) (Van der Maaten & Hinton, 2008). It transforms pairwise similarities of data points to probability distributions of the high-dimensional data as well as the low-dimensional embedding and then minimizes the Kullback-Leibler divergence between them. While the original t-SNE is based on the gaussian kernel function, equation 3.1, we replace it with the tangent space dissimilarity measure defined in equations 3.3 and 3.4. Since high-dimensional data spaces are typically sparse, we use a k-nearest neighbors instead of radius neighborhood, and we set $p = 100\%$ (i.e., we normalize the distances in equation 3.3 by dividing them by their maximum values). Furthermore, we compare the results to the original t-SNE using the Euclidean distance to compute the neighborhood probabilities and a t-SNE version using the Mahalanobis distance instead. The latter is based on the covariance of the data and can therefore align with global directions of major variance. Therefore, the locally aligned formulation, in equations 3.3 and 3.4, is compared to "no alignment" and "global alignment."

We demonstrate the strategy using two real nonlinear dimensionality-reduction benchmark data sets: (1) COIL20 which consists of images (with $32 \times 32 = 1024$ pixels) of 20 objects rotated 72 times (5 degrees per image) forming one-dimensional manifolds, and (2) USPS, which contains 9298 gray-scale images (with 16×16 pixels) of handwritten digits. As a preprocessing step, we apply PCA to decrease the dimensionality to 20. The different t-SNE outputs are compared and evaluated using the quality of group compactness (QGC) (Gorban & Zinovyev, 2010) as a measure of how close samples from the same class remain in the embedding space. Let $c(i; k)$ denote the number of points in the k-neighborhood of $\boldsymbol{x}_i$ with the same label; then the compactness of a class is defined as

$$QGC_k(l) = \frac{1}{k \cdot N(l)} \sum_{y_i = l} c(i; k),$$

where $N(l)$ is the number of samples with the label l. If the QGC for a class is close to one, it means the class is compact and well separated from others. Since t-SNE is nonconvex, it generates different results for each run, and hence we repeat the experiment 10 times and report the average quality.

Figure 6 shows example t-SNE embeddings for COIL20 and USPS (panels a and c), accompanied by results using equation 3.3 with $d = 1$ and

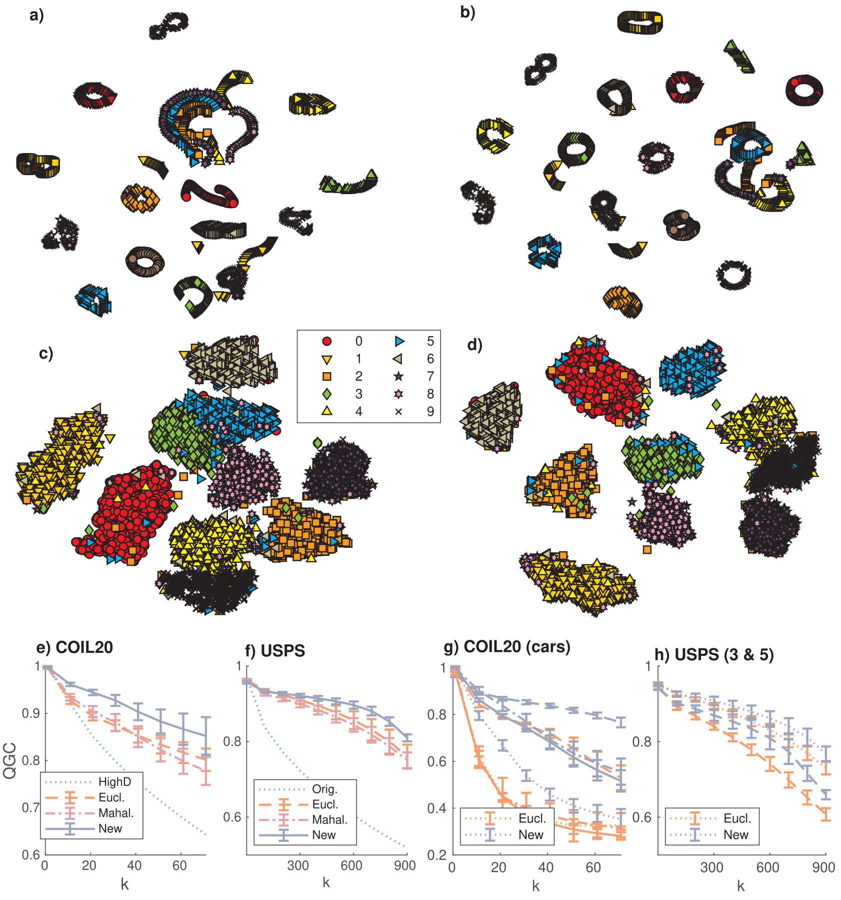

Figure 6: The impact of the new similarity measures on the t-SNE. Top row, COIL20: (a) using original t-SNE (perpl. 20) and (b) tSNE+ using equation 3.3 ($d = 1$, $k = 10$). Middle row USPS: (c) using original t-SNE (perpl. = 30) and (d) tSNE+ using equation 3.4 ($k = 20$). Bottom row: Average QGC measures for COIL20 and USPS (panels e and f). (g) QGC curves for four classes of toy cars in COIL20 and (h) of digits 3 and 5 in USPS.

$k = 10$ and equation 3.4 with $k = 20$ (panels b and d). Panels e and f depict the average QGC curves for the different similarity measures, respectively. It can be seen that the new similarity measures outperform others, especially for COIL20, since we use our knowledge about the intrinsic dimensionality $d = 1$ of the manifolds. For the USPS, it is clear that it does a better job for more global structures, with clusters visible for $k > 200$. Since the new weights only consider the distance to the estimated manifold, it can recover more closed loops with less distortion (see panels a and b). Moreover, the four objects corresponding to four types of toy cars cannot be separated using the original t-SNE because it maps cars with the same orientations close to each other. As seen in panel g, the new weights do a better job in embedding them. Not knowing the intrinsic dimensionality of manifolds in the handwritten digit data set, we use equation 3.4 to define a similarity measure for t-SNE. Although t-SNE with the gaussian kernel function can recover most classes, a strong overlap between samples from digits 3 and 5 is visible. The new weights, however, successfully separate them and increase the QGC measure, as seen in panels d and h.

6.3 Manifold Alignment Aware Ants. In this section, we analyze the capability of M3A in recovering structures surrounded by background noise. First, we compare it to related state-of-the-art techniques for denoising low-dimensional manifolds. And second, we demonstrate the performance of our algorithm on a real-world astronomical data set, the GAIA DR2 catalog (Gaia Collaboration et al., 2018), to extract stellar structures.

6.3.1 Extracting Low-Dimensional Manifolds. Typically, manifold learning assumes that data points are lying on a low-dimensional manifold embedded in a higher-dimensional space. However, in practice, manifolds are disrupted by high-dimensional noise, for example, stemming from equipment or the presence of samples that do not belong to the manifolds. In those cases, the goal is to extract the manifolds by removing the background noise and simultaneously suppressing the noise level on the manifolds. To compare the performance of denoising techniques, we create a synthetic 1D manifold forming a circle ($r = 6$), which misses a part of its arc. Furthermore, we randomly select 3000 points from the manifold and add gaussian noise $N(0, 0.3)$. To model background noise, 3000 samples are generated following a uniform distribution in the square $[-15, 15] \times [-10, 20]$ (see Figure 7a).

In the following, we compare four denoising techniques to the proposed M3A. We use (1) MD with hyperparameter settings $\delta t = 0.1$, $N_{\text{iter}} = 20$; and $k = 40$; (2) MBSM with $k = 40$ and $N_{\text{iter}} = 40$; and (3) LLD with $\lambda = 0.01$, $k = 40$ and $\sigma = 1$. Finally, LLPD is deployed using a threshold of 0.15. Moreover, in order to select r in M3A, we follow a two-step strategy. First, we apply the algorithm using $r = 3$ to remove the background noise. Note that since the density of the background is lower than the manifold, it can be done using a wide range of r. Then we use the strategy explained in section 6.1 to find the best r for the remaining points.

Some techniques, such as MD, MBMS, and LLD, are not designed to deal with the presence of the background noise and are therefore not directly suitable for the given situation, as shown in Figure 7. Since the manifold denoising (MD) algorithm pushes samples toward dense regions, it has two disadvantages: first, it creates some new dense structures in data

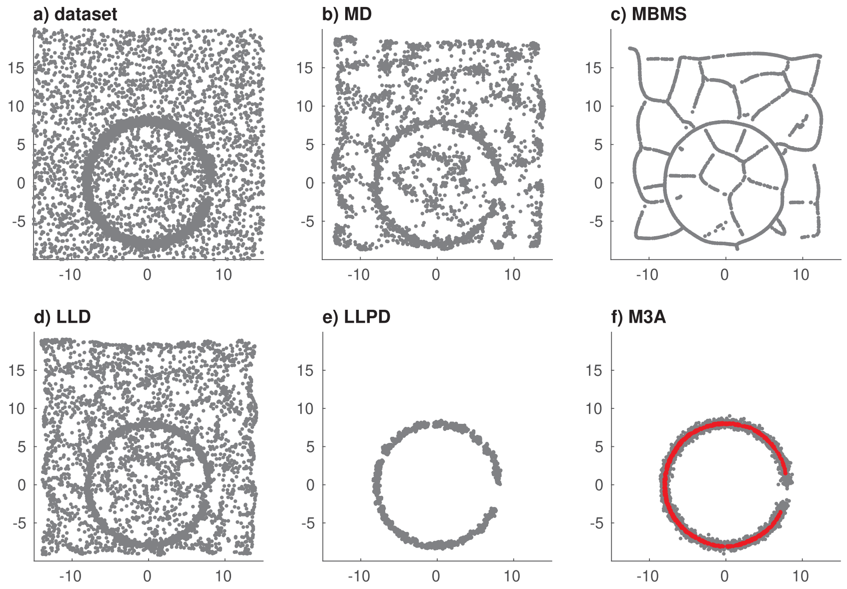

Figure 7: (a) 1D circular manifold embedded in uniform background noise. Best results of (b) MD, (c) MBMS, (d) LLD, and (e) LLPD (containing 2000 samples, i.e. 35%). (f): Denoised M3A result where gray and red points show 2700 (45%) and 780 (%13) samples with $r = 3$ and $r = 2$.

space, and second, the real manifold becomes discontinuous (see Figure 7b). Since manifold blurring mean shift (MBMS) limits the movement of samples to be parallel to the manifold normals, it can prevent discontinuity on the manifold. However, it creates many artificial 1D manifolds and fails to recover the missing arc in the circle structure of interest (see Figure 7c). Similar to MBMS, the presence of background noise highly influences the result of the LLD algorithm, which also fails to recover the manifold (see Figure 7d). In contrast to previous methods, LLPD considers the presence of the background noise and extracts the samples in dense regions recovering the noisy circle. Although it does not create new structures, it cannot denoise the manifold. It also fragments the structure into smaller clusters (see Figure 7e). On the other hand, our method encourages random walkers to stay close to the manifold and overcomes all three problems: (1) it recovers the circle from the background noise, (2) it preserves the missing part in the arc without creating artificial structures, and (3) it reduces the manifold noise by subsampling based on the highest pheromone level (see Figure 7f).

6.3.2 Detecting Dense Structures in Real-World Data. An important task in astronomy is to extract stellar structures, such as galaxies and globular

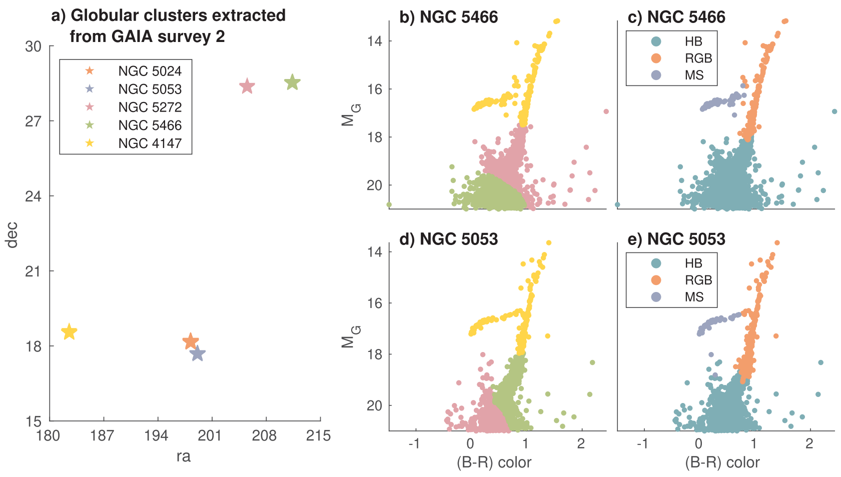

Figure 8: Five clusters formed by 0.02% of samples with highest pheromone values resulting from M3A ($r = 0.1$, $N_{\text{steps}} = 5 \times 10^6$) applied to a window in the sky between right ascension (ra) $\in]180, 220[$ and declination (dec)$]15, 30[$. (b–e) The application of spectral clustering on the photometric data of two globular clusters (b,d) Using gaussian kernel as a similarity measure. (c,e) Using the transition probability in equation 4.2 ($r = 0.5$).

clusters. Here, we apply M3A to a part of the GAIA DR2 catalog (Gaia Collaboration et al., 2018) containing 1,071,714 light sources. The data set contains seven features, including five denoting positional information (right ascension (ra), declination (dec), motion along ra and dec, and parallax), as well as photometric information, that is, G-band magnitude and (B-R) color. As a preprocessing step, we normalize the data set and select the k nearest neighbor (with $k = 20$) to speed up the computation. First, we use the positional information to detect stellar structures. Figure 8a shows that M3A detects five of the known globular clusters (GC): (1) NGC4147, (2) NGC5024, (3) NGC5053, (4) NGC5272, and (5) NGC5466. Second, we would like to cluster the members of a GC. To extract stars of a GC, we use objects that are not farther than 0.05 degree from the center of the GC. Then we use their photometric information since it reveals the stage of life a star is in (for more details, see Mohammadi, Petkov, Bunte, Peletier, & Schleif, 2019). Figures 8b and 8d display the clusters detected by spectral clustering using a gaussian kernel (Von Luxburg, 2007). However, the resulting clusters do not correlate to any astrophysical meaning. If M3A is applied on the extracted light sources, we obtain the transition probability matrix P. After symmetrization $P' = \frac{1}{2}(P + P^T)$, the spectral clustering can be implemented based on P'. From Figures 8c and 8e, we see that the resulting clusters coincide with the three known groups: Main Sequence (MS), Horizontal Branch (HB), and

Red Giant Branch (RGB) (see Mohammadi et al., 2019). It can be seen that MS is separable since it is a 2D manifold while HB and RGB are 1D manifolds. Moreover, since HB is almost a line perpendicular to RGB, the weight values between these two clusters are small, and as a result, they are separable. In summary, M3A provides (1) a pheromone distribution that can be used to find stellar structures and (2) a transition probability matrix that can be used to group stars according to the stage of their life.

6.4 Improved Density Estimation. A common task in machine learning is to estimate the underlying probability density function (pdf) given a limited number of data points. A typical way to model it is to use a finite mixture model,

$$\hat{f}(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{K} p(k)\hat{f}(\boldsymbol{x};\theta_k),$$

where $p(k)$ and θ_k are the mixture weight and the parameter values of the kth component, respectively, while $\boldsymbol{\theta}$ denotes the set of all parameters. If the number of models (K) is much smaller than the number of samples (N), it provides a sparse representation for the data set. A well-known algorithm is the gaussian mixture model (GMM) (Bishop, 2006), which uses a mixture of full-rank gaussian distributions to model the pdf. Its parameters are learned via optimizing the likelihood function through an iterative process, called expectation-maximization (EM). It is known that EM is sensitive to the initialization step, and it may get stuck in local optima. Moreover, the time needed for EM is dependent on the number of components and increases for large data sets with many samples and dimensions. One way to avoid this problem is to use a nonparametric method, such as Parzen window (PW), which is a special case of the finite mixture model with $K = N$, and it has only one parameter σ,

$$\hat{f}(\boldsymbol{x};\sigma) = \frac{1}{N}\sum_{k=1}^{N} \mathcal{K}(\boldsymbol{x};\boldsymbol{x}_k,\sigma),$$

where $\mathcal{K}$ is the gaussian kernel function. However, if the data points are distributed along a low-dimensional manifold, the spherical gaussian kernel is often not an optimal choice (Vincent & Bengio, 2003). Therefore, Vincent and Bengio (2003) proposed the manifold Parzen window (MPW), which allows the gaussian distributions to have elliptical shapes instead of spherical. Whereas PW and MPW may provide a reliable estimation, both suffer from high computational costs in the test phase due to the large number of gaussian models. To prevent this problem, several methods, such as simplifying mixture models (SMM) (Zhang & Kwok, 2010) and hierarchical

Algorithm 3: Partitioning the Data Space.

```
Result: set $\mathcal{S}$, containing the center of balls;
1 Input: The pheromone vector $\mathbf{f} = [f_1, f_2, ..., f_N]$ and the balls' radius $r_{\text{ball}}$;
2 Set $\mathcal{D} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, $\mathcal{S} = \phi$;
3 while $\mathcal{D} \neq \phi$ do
4     $l = \arg\max_{\boldsymbol{x}_i \in \mathcal{D}} f_i$;
5     find $l$'s neighbors $\mathcal{N}_l = B_{r_{\text{ball}}}(\boldsymbol{x}_l)$;
6     $\mathcal{S} \leftarrow \mathcal{S} \cup \{l\}$;
7     $\mathcal{D} \leftarrow \mathcal{D} - \{l\} - \mathcal{N}_l$;
8 end
```

clustering of a mixture model (HCMM) (Goldberger & Roweis, 2005), were proposed. They start with a large mixture model and then construct a simpler model, such that the distance between the original model and the simplified one is minimized. In contrast to these approaches, Wang, Tino, Fardal, Raychaudhury, and Babul (2009) proposed a new method, fast Parzen window (FPW), which partitions the data space via hyperballs positioned randomly with fixed radii r_{ball}. Then it fits a full-rank gaussian distribution for each ball only. Extending the latter idea, we use the pheromone values obtained by M3A to find the best position for the center of the hyperballs. We start with the whole data set $\mathcal{D}$ and pick the point x_l with the highest pheromone as the center of a ball. Then we remove all points within the ball from $\mathcal{D}$. Then we continue picking the next point in $\mathcal{D}$ with the highest pheromone as the new center and remove its neighbor from $\mathcal{D}$. Following this strategy, we obtain a more compact list of centers. Its pseudocode is summarized in algorithm 3.

To demonstrate the performance of the new strategy, we use three data sets:

1. A synthetic spiral shape manifold with gaussian noise $N(0, 0.04)$ for which 1000 samples are generated as a training set (see Figure 9a) and another 20000 samples as a test set.
2. Two intersected circular manifolds with radius 2. (a) In the above circle, the density varies, and the noise level is 0.2. (b) At the bottom, the noise level varies between 0 to 0.3 (see Figure 10a). We generated 3350 and 16,750 points for training and testing, respectively.

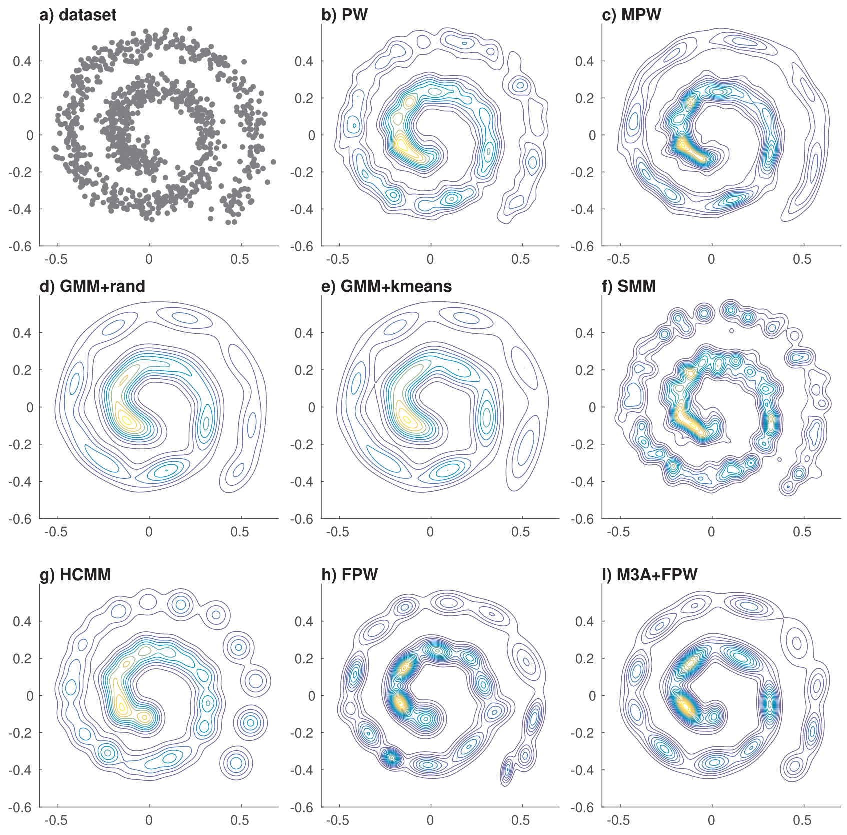

Figure 9: (a) The noisy spiral data set. (b–h) The contour curves for the density estimators: PW, MPW, GMM+rand, GMM+kmeans, SMM, HCMM, FPW, and (l) M3A+FPW: neighborhood radius $r_{rw} = 0.13$ and balls' size $r_{\text{ball}} = 0.21$.

3. An astronomical simulation of a jellyfish galaxy that contains 3D position information of 58,531 particles.

To evaluate the performance, we use the average log likelihood (ALL) to measure how well the adapted model can describe the data set. We furthermore use 10-fold cross-validation on the training set to find the best hyperparameter values.

Table 1 reports the results of the above algorithms on the spiral shape manifold. From cross-validation, hyperparameters are found where m, k, and σ are the number of components and neighbors and the scale of gaussian kernel, respectively, and r_{rw} is the neighbor size for M3A. Note that GMM^k is the gaussian mixture model when k-means is used to initialize components. We repeat the experiment for M3A 10 times. While ALL for

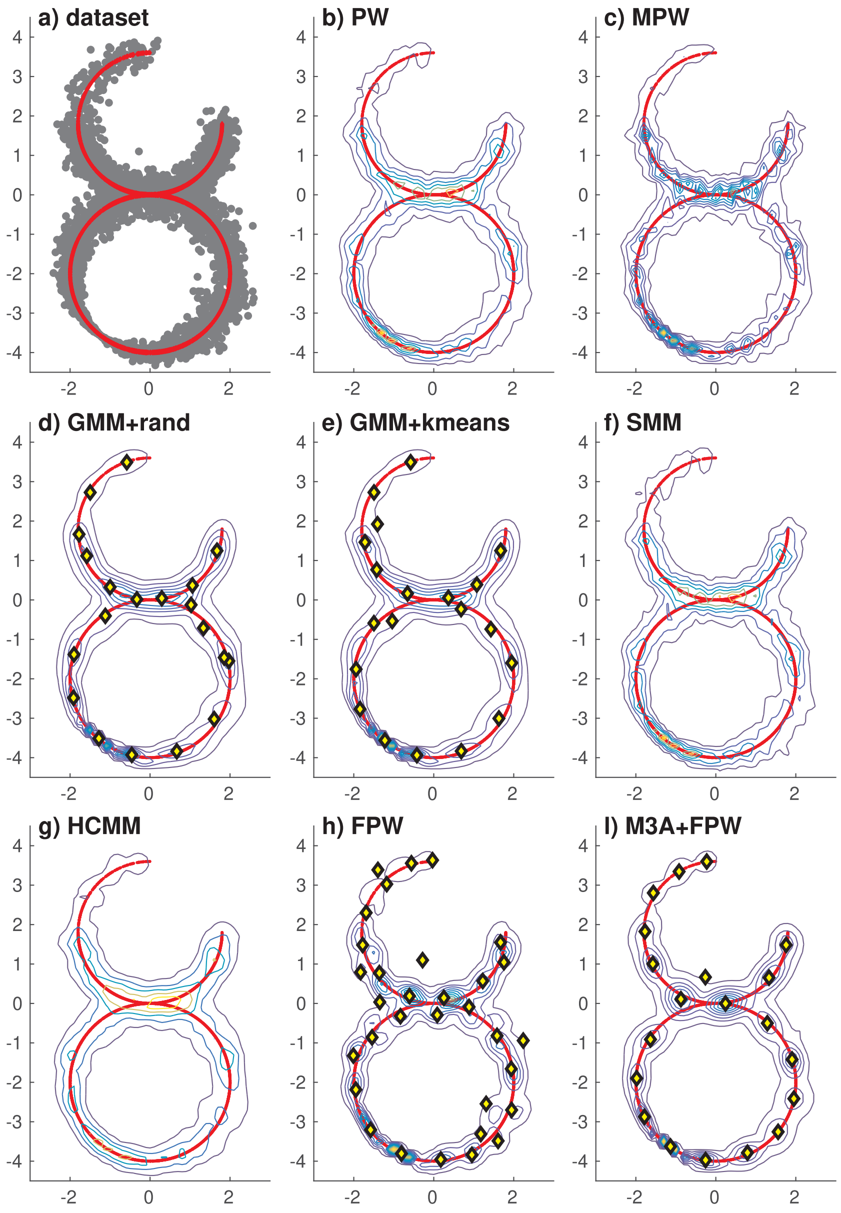

Figure 10: (a) Two circular manifolds with variation in density and noise level. (b–l) The contour curves for the compared eight density estimators. Note that black diamonds represent the means of the gaussian distributions.

Table 1: Comparison of Density Estimators for the Spiral Data Set.

Algorithm	Parameters	ALL	(std)
PW	$\sigma = 0.03$	0.3130	(0.0)
MPW	$k = 45$	0.3003	(0.0)
FPW	$r_{\text{ball}} = 0.18$ ($m = 20.8$)	0.2851	(0.0092)
SMM	$m = 95$	0.2762	(0.0116)
HCMM	$m = 35, \sigma = 0.06$	0.2832	(0.0049)
GMM	$m = 20$	0.2944	(0.0028)
GMM^k	$m = 15$	0.2802	(0.0049)
M3A+FPW	$r_{rw} = 0.13, r_{\text{ball}} = 0.21$ ($m = 17.1 \pm 1$)	0.3101	(0.0052)

Table 2: Comparison of Density Estimators for the Two Circular Manifolds.

Algorithm	Parameters	ALL	(std)
PW	$\sigma = 0.09$	−2.668	(0.0)
MPW	$k = 140$	−2.689	(0.0)
FPW	$r_{\text{ball}} = 0.75$ ($m = 31$)	−2.682	(0.0116)
SMM	$m = 800$	−2.671	(0.0007)
HCMM	$m = 130, \sigma = 0.15$	−2.700	(0.0014)
GMM	$m = 20$	−2.620	(0.0036)
GMM^k	$m = 20$	−2.621	(0.0034)
M3A+FPW	$r_{rw} = 0.6, r_{\text{ball}} = 0.95$ ($m = 20.6 \pm 1$)	−2.650	(0.0027)

the M3A+FPW (i.e., algorithm 3) is comparable with PW, it provides a much sparser representation for the data set and speeds up the computation in the testing phase. Moreover, Figure 9 shows the contour curves of the pdfs. We observe that most methods represent the center of the spiral fairly well, with the exception of PW and SMM, which provide the noisiest model. Strikingly, PW, SMM, and HCMM discontinue the manifold in the less dense tail on the right side of the spiral, dividing it into several small clusters. Our strategy M3A+FPW, on the other hand, not only provides a compact model but also successfully tracks the underlying structure and stays very close to the original manifold as compared to others.

In Table 2 we present the outputs of the studied algorithms on the two circular manifolds. As in the previous examples, cross-validation is used to tune hyperparameters, and then the performance of the algorithms on the test set is reported. While the FPW+M3A yields a sparse representation for the data, it also outperforms (in terms of ALL) others, except for the GMM methods. However, since the M3A can be parallelized, it is more suitable for big data sets, in comparison to GMM techniques. Moreover, we display the contour curves of pdfs in Figure 10 where the true manifold (without noise) is shown in red. It can be seen that the M3A helps FPW to be more compact and more successful in tracking the manifolds, as indicated by the

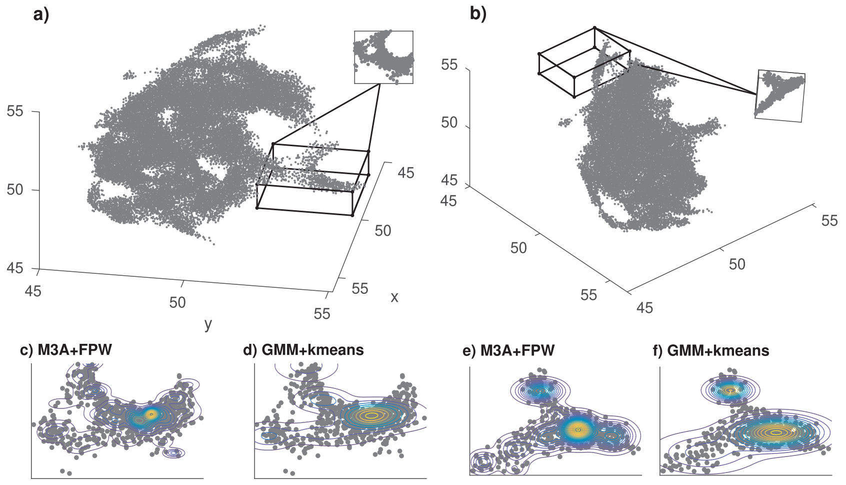

Figure 11: The simulated jellyfish galaxy (viewpoint a and b). (c–f) Contour curves of density estimators projected by the first two principal directions of gaussian covariance matrices for zoomed regions a (left) and b (right) using algorithm 3 (c,e) and GMM initialized by k-means (d,f).

black diamond in the plot. In comparison to GMMs, FPW+M3A uniformly distributes gaussian models on the manifolds, even in less dense regions. It is especially important in some applications such as astronomy where low-density streams have high importance in studying the evolution of astronomical structures. Therefore, similar to previous experiments, we conclude that FPW+M3A builds a compact model, which is more successful in following the manifolds, regardless of their densities.

Astronomical simulations are often used to study the evolution of astronomical objects through time. Since a simulation often includes many millions of particles, it is not possible to track each particle individually. Therefore, it is desirable to encode the distribution of particles via a sparse pdf and follow its evolution instead. Since our goal in this section is to compare different density estimators, we only use one time snapshot of a simulated jellyfish galaxy, as shown in Figures 11a and 11b. This data set is challenging since the density changes significantly in the data space. While the head of the jellyfish is very dense, the outer parts depict much lower density. Those parts contain some low-dimensional structures called streams, as highlighted by boxes in Figures 11a and 11b. During the evolution process, these structures are more affected by other astronomical objects and change over time. Thus, in addition to the dense regions, a suitable density estimator should capture these lower density regions well.

Table 3: Comparison of Density Estimators for the Simulated Jellyfish Galaxy.

Algorithm	Parameters	ALL	(std)	d_H	(std)	time in s	(std)
PW	$\sigma = 0.08$	−3.775	(0.018)	0.264	(0.040)	0.2	(0.0)
FPW	$r = 0.4, m = 949$	−3.855	(0.023)	0.633	(0.071)	9.2	(0.4)
SMM	$m = 4100$	−3.815	(0.021)	0.699	(0.115)	144.8	(10.5)
SMM	$m = 600$	−4.249	(0.032)	0.393	(0.053)	63.0	(5.2)
HCMM	$\sigma = 0.17, m = 600$	−3.967	(0.014)	0.697	(0.091)	12.4	(0.5)
GMM	$m = 300$	−3.772	(0.025)	2.916	(0.600)	1244.4	(257.4)
GMM^k	$m = 300$	−3.767	(0.024)	3.060	(0.654)	909.3	(63.1)
M3A+FPW	$r_{rw} = 0.4, r_{\text{ball}} = 0.5, m = 566$	−3.838	(0.025)	0.737	(0.109)	165.5	(2.1)
M3A+FPW	$r_{rw} = 0.4, r_{\text{ball}} = 0.65, m = 309$	−3.882	(0.024)	0.823	(0.072)	151.2	(11.9)

We compare our M3A+FPW algorithm with the result of GMM initialized by k-means visually and additionally report the quality of all density estimators as before.

From Table 3 it can be seen that the GMM outperforms the other methods in terms of ALL, which is expected since it aims to optimize the likelihood function. However, its optimization process is time-consuming for big data sets, which makes it challenging to be applied in big simulations, as indicated by large training times reported in the last column of Table 3.[9] PW as before provides an unnecessary complex model, which is not very desirable for this application. Alternatively, the M3A+FPW strategy presents a sparse model while preserving a comparable ALL.

In addition to ALL we report the Hausdorff distance,

$$d_H(A, B) = \max\left\{ \max_{a\in A} \min_{b\in B} d(a, b), \max_{b\in B} \min_{a\in A} d(a, b) \right\},$$

where d denotes the Euclidean distance and A and B are the set of particles and the set of centers of gaussian models, respectively.

It is informative since it shows how much the model captures the original low-density structures. To compare our method to GMM, we increase the balls' radius to $r_{\text{ball}} = 0.65$. From the table, it is clear that the Hausdorff distance of the proposed method is much lower than the one achieved by GMM. To visually see its success, we added the contour plots (see panels c–f). It can be seen that the new method is more successful in fitting the pdf along with the low-dimensional structure. In summary, if selecting a model is a trade-off between simplicity (sparsity) and capturing structural details,

[9] Here, we distribute the M3A ants among 10 processors (i.e., 5 ants per CPU) and restrict the number of step in each round to $n = 10{,}000$.

the M3A+FPW strategy, outlined in algorithm 3, provides a sparse model with a very competitive ALL value.

7 Conclusion

In some applications, such as astronomy, it is common to have low-dimensional structures buried inside big data sets. Therefore, it is desirable to have a method that extracts these structures of varying density while their continuity is kept unchanged. Although there are several methods, they often fail to either extract manifolds or keep their continuity. Inspired by the ant colony algorithm, we propose a new method where a value, called pheromone, is assigned to each sample. Later, these quantities can be used to reveal manifolds without the undesirable effects. To study the behavior of the algorithm and the effect of its hyperparameters, we provide a theoretical analysis using the Markov chain framework, where we consider a noisy manifold and apply the algorithm examining two cases: a random walker with an unbounded number of steps and an unbounded number of walkers with a fixed number of steps. We show for both cases that the pheromone distribution captures information about the distances of data points to the underlying manifold. In addition to the theory, we empirically analyze the algorithm using synthetic and real data sets, demonstrating three different scenarios: denoising and clustering manifolds, visualizing data, and density estimation. In all investigated scenarios and application examples, strategies based using M3A exhibit comparable or superior results in suppressing the noise, capturing the manifolds, and providing sparser models.

Acknowledgments

This work was supported by the European H2020-MSCA-ITN SUrvey Network for Deep Imaging Analysis and Learning (SUNDIAL), project ID 721463. We thank the Center for Information Technology of the University of Groningen for its support and for providing access to the Peregrine high-performance computing cluster. Furthermore, this work made use of data from the European Space Agency (ESA) mission Gaia (https://www.cosmos.esa.int/gaia), processed by the Gaia Data Processing and Analysis Consortium (DPAC, https://www.cosmos.esa.int/web/gaia/dpac/consortium). Funding for the DPAC has been provided by national institutions, in particular the institutions participating in the Gaia multilateral agreement.

References

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*(6), 1373–1396. 10.1162/089976603321780317

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 509–517. 10.1145/361002.361007

Berry, T., & Sauer, T. (2016). Local kernels and the geometric structure of data. *Applied and Computational Harmonic Analysis*, *40*(3), 439–469. 10.1016/j.acha.2015.03.002

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Berlin: Springer.

Blum, C., Roli, A., & Dorigo, M. (2001). HC–ACO: The hyper-cube framework for ant colony optimization. In *Proceedings of the Metaheuristics International Conference*, vol. 2 (pp. 399–403).

Chu, S.-C., Roddick, J. F., Su, C.-J., & Pan, J.-S. (2004). Constrained ant colony optimization for data clustering. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence* (pp. 534–543). Berlin: Springer.

Cinlar, E. (2013). *Introduction to stochastic processes*. North Chelmsford, MA: Courier Corporation.

Coifman, R. R., & Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis*, *21*(1), 5–30. 10.1016/j.acha.2006.04.006

Dixit, P. D. (2019). Introducing user-prescribed constraints in Markov chains for nonlinear dimensionality reduction. *Neural Computation*, *31*(5), 980–997. 10.1162/neco_a_01184, PubMed: 30883279

Donoho, D. L., & Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. In *Proceedings of the National Academy of Sciences*, *100*(10), 5591–5596. 10.1073/pnas.1031596100

Dorigo, M. (1992). *Optimization, learning and natural algorithms*. PhD diss., Politecnico di Milano.

Dorigo, M., Maniezzo, V., & Colorni, A. (1991). *Positive feedback as a search strategy* (Technical Report 91-016). Politecnico di Milano, Italy.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *26*(1), 29–41. 10.1109/3477.484436

Gaia Collaboration, Brown, A., Vallenari, A., Prusti, T., de Bruijne, J., Babusiaux, C., et al. (2018). Gaia data release 2: Summary of the contents and survey properties. *Astronomy and Astrophysics*, *616*, 1–22. 10.1051/0004-6361/201833051

Gil, C., Baños, R., Ortega, J., Márquez, A. L., Fernández, A., & Montoya, M. (2011). Ant colony optimization for water distribution network design: A comparative study. In *Proceedings of the International Work-Conference on Artificial Neural Networks* (pp. 300–307). Berlin: Springer.

Goldberger, J., & Roweis, S. T. (2005). Hierarchical clustering of a mixture model. In Y. Weiss, B. Schölkopf, & J. Platt (Eds.), *Advances in neural information processing systems, 18* (pp. 505–512). Cambridge, MA: MIT Press.

Golub, G. H., & Van Loan, C. F. (2012). *Matrix computations*. Baltimore, MD: Johns Hopkins University Press.

Gong, D., Sha, F., & Medioni, G. (2010). Locally linear denoising on image manifolds. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (pp. 265–272).

Gorban, A. N., & Zinovyev, A. (2010). Principal manifolds and graphs in practice: From molecular biology to dynamical systems. *International Journal of Neural Systems*, *20*(3), 219–232. 10.1142/S0129065710002383, PubMed: 20556849

Hein, M., & Maier, M. (2006). Manifold denoising. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems*, *19* (pp. 561–568). Cambridge, MA: MIT Press.

Kaslovsky, D. N., & Meyer, F. G. (2014). Non-asymptotic analysis of tangent space perturbation. *Information and Inference: A Journal of the IMA*, *3*(2), 134–187. 10.1093/imaiai/iau004

Klicpera, J., Weissenberger, S., & Günnemann, S. (2019). Diffusion improves graph learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems, 32* (pp. 13333–13345). Red Hook, NY: Curran.

Kulkarni, V. G. (1999). *Modeling, analysis, design, and control of stochastic systems*. Berlin: Springer.

Lerman, G., McCoy, M. B., Tropp, J. A., & Zhang, T. (2015). Robust computation of linear models by convex relaxation. *Foundations of Computational Mathematics*, *15*(2), 363–410. 10.1007/s10208-014-9221-0

Little, A. V., Maggioni, M., & Murphy, J. M. (2020). Path-based spectral clustering: Guarantees, robustness to outliers, and fast algorithms. *Journal of Machine Learning Research*, *21*, 1–66. 34305477

Little, A. V., Maggioni, M., & Rosasco, L. (2017). Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. *Applied and Computational Harmonic Analysis*, *43*(3), 504–567. 10.1016/j.acha.2015.09.009

Maniezzo, V. (1999). Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *INFORMS Journal on Computing*, *11*(4), 358–369. 10.1287/ijoc.11.4.358

Mohammadi, M., & Bunte, K. (2020). Multi-agent based manifold denoising. In *Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning* (pp. 12–24). Berlin: Springer.

Mohammadi, M., Petkov, N., Bunte, K., Peletier, R. F., & Schleif, F.-M. (2019). Globular cluster detection in the Gaia survey. *Neurocomputing*, *342*, 164–171. 10.1016/j.neucom.2018.10.081

Rizzoli, A. E., Oliverio, F., Montemanni, R., & Gambardella, L. M. (2004). Ant colony optimisation for vehicle routing problems: From theory to applications. *Galleria Rassegna Bimestrale Di Cultura*, *9*(1), 1–50.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*(5500), 2323–2326. 10.1126/science.290.5500.2323, PubMed: 11125150

Runkler, T. A. (2005). Ant colony optimization of clustering models. *International Journal of Intelligent Systems*, *20*(12), 1233–1251. 10.1002/int.20111

Stützle, T., & Hoos, H. H. (2000). Max–min ant system. *Future Generation Computer Systems*, *16*(8), 889–914.

Tsai, C.-F., Tsai, C.-W., Wu, H.-C., & Yang, T. (2004). ACODF: A novel data clustering approach for data mining in large databases. *Journal of Systems and Software*, *73*(1), 133–145. 10.1016/S0164-1212(03)00216-4

Tu, L. W. (2011). *An introduction to manifolds*. Berlin: Springer.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11), 2579–2605.

Vincent, P., & Bengio, Y. (2003). Manifold Parzen windows. In S. Becker, S. Thrun, & K. Overmayer (Eds.), *Advances in neural information processing systems, 16* (pp. 849–856). Cambridge. MA: MIT Press.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, *17*(4), 395–416. 10.1007/s11222-007-9033-z

Wang, W., & Carreira-Perpinán, M. A. (2010). Manifold blurring mean shift algorithms for manifold denoising. In *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 1759–1766). Piscataway, NJ: IEEE.

Wang, X., Tino, P., & Fardal, M. A. (2008). Multiple manifolds learning framework based on hierarchical mixture density model. In *Proceedings of the European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases* (pp. 566–581). Berlin: Springer.

Wang, X., Tino, P., Fardal, M. A., Raychaudhury, S., & Babul, A. (2009). Fast Parzen window density estimator. In *Proceedings of the 2009 International Joint Conference on Neural Networks* (pp. 3267–3274). Piscataway, NJ: IEEE.

Zhang, K., & Kwok, J. T. (2010). Simplifying mixture models through function approximation. *IEEE Transactions on Neural Networks*, *21*(4), 644–658. 10.1109/TNN.2010.2040835, PubMed: 20181542

Zhang, Z., & Zha, H. (2003). Nonlinear dimension reduction via local tangent space alignment. In *Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning* (pp. 477–481). Berlin: Springer.

Received February 4, 2021; accepted November 4, 2021.