Point Cloud Denoising Algorithm via Geometric Metrics on the Statistical Manifold

Xiaomin Duan, Li Feng and Xinyu Zhao

Abstract: A denoising algorithm was proposed for point cloud with high-density noise. The algorithm utilized geometric metrics on the statistical manifold and applied the idea of clustering K-means based on local statistical characteristics between noise and valid data. First, by calculating the expectation and covariance matrix of the data points, the point cloud with high-density noise was projected onto the Gaussian distribution family manifold, aiming to form the parameter point cloud. Afterwards, the geometry metrics were assigned to the manifold, and the K-means algorithm was applied to cluster the parameter point cloud, so as to classify the valid data and noise. Furthermore, in order to analyze the robustness of the means with different metrics, the approximate values of their influence functions were calculated, respectively. Finally, simulation analysis was conducted using the algorithm based on geometric metrics to verify the effectiveness in point cloud denoising.

Keywords: point cloud denoising; statistical manifold; differential geometry; influence function

1. Introduction

Point cloud denoising is a widely used data processing technique in various fields, such as 3D reconstruction, indoor positioning systems and positioning, intelligent manufacturing, virtual reality and augmented reality, and geological exploration and seismic research [1]. With the ability to improve the quality and accuracy of point cloud data and provide a reliable data foundation for other applications. Point cloud denoising aims to effectively eliminate noise points, smooth the reconstructed surface model, and maintain the original topology and geometric characteristics of the sampled surface. When the density of a point in a local area of the point cloud is significantly higher or lower than that of the surrounding points, it is referred to as high-density noise or low-density noise, respectively. The presence of high-density noise can be determined by calculating the density or number of points in the local neighborhood using methods such as nearest neighbor distance or K-nearest neighbor. In specific applications, thresholds for high-density and low-density noise can be established through experiments and optimizations based on actual requirements.

The popular point cloud processing platform 'Point Cloud Library' offers denoising methods such as radius outlier removal and statistical outlier removal, but these are only suitable for low-density noise data [1]. Recently, a new approach using Wasserstein curvature has been proposed for point cloud denoising; however, it mistakenly identifies some real information in flat areas as noise [2].

The K-means algorithm is known for its low time complexity and fast running speed, making it suitable for processing most continuous variable data [3]. It performs well in dealing with data sets that have spherical clustering structures and can complete clustering in a short period of time. However, the traditional K-means algorithm only considers Euclidean distance when calculating the similarity between samples and does not account for the statistical structure of point cloud. This limitation makes it challenging for the
algorithm to distinguish between random noise and valid data [3,4]. To address this issue, a clustering algorithm is proposed based on the neighborhood density of each data in [5,6]. On the manifold, the local geometric structure of noise and valid data are inconsistent, and thus the data with similar local geometric characteristics need to be considered as the same cluster. In [7,8], a K-means clustering algorithm is used to denoise point cloud, but only a few metrics are adopted to calculate the distance on the manifold. Meanwhile, the influence function, which aims to evaluate the robustness of the mean induced by each metric to outliers, is not analyzed in these studies.

This manuscript extends our previous ideas in [8] by proposing a point cloud denoising method based on the geometry of the Gaussian distribution family manifold. The manifold is endowed with five metrics, including the Euclidean metric, the affine-invariant Riemannian metric, the log-Euclidean metric, the Kullback–Leibler divergence as well as the symmetrized Kullback–Leibler divergence [9]. To evaluate the robustness of the mean endowed by each metric to outliers, the influence functions of geometric metrics are calculated. As stated in [10], it can be very difficult to solve the matrix equations for the influence functions directly, so we calculate the approximate values of these influence functions. To evaluate the denoising effects of these metrics, the true positive rate (TPR), the false positive rate (FPR), and the signal–noise rate growing (SNRG) are adopted. TPR refers to the proportion of correctly predicted positive cases among all actual positive cases. A higher TPR means less real data loss. In fact, higher TPR, higher SNRG, and lower FPR indicate that the algorithm can better distinguish between real data and noise. Our proposed algorithm will be evaluated by the above three standards.

The contributions in this paper are summarized as follows.

1. A K-means clustering algorithm is proposed to denoise point cloud with high-density noise by leveraging the difference in local statistical characteristics between noise and valid data. The algorithm operates on a Gaussian distribution family manifold.
2. By calculating the expectation and the covariance matrix of the data points, we map the original point cloud onto the Gauss distribution family manifold to form the parameter point cloud. Next, the metrics are assigned to the manifold, and the K-means method is applied to cluster the parameter point cloud, aiming to classify the original point cloud.
3. With the purpose of analyzing the robustness of the means with different metrics, the approximate values of their influence functions are calculated, respectively. The simulation results demonstrate that using geometry metrics yields better denoising effects compared to using the Euclidean metric. Additionally, mean influence functions with geometry metrics exhibit greater robustness than that with the Euclidean metric.

The rest of the current work is structured as follows. Briefly, Section 2 introduces the Riemannian framework of the Gaussian distribution family manifold. Section 3 proposes a K-means clustering algorithm to denoise point cloud with high-density noise and calculates the approximate values of their influence functions with different metrics. In addition, Section 4 presents the effectiveness of the proposed algorithm to denoise point cloud and verifies our obtained properties of influence functions.

2. Preliminaries

In this study, the set of all \( n \times n \) symmetric matrices can be indicated:

\[
S(n) = \{ A \in \mathbb{R}^{n \times n} | A^T = A \}. \tag{1}
\]

Moreover, the set of all \( n \times n \) symmetric positive-definite matrices can be indicated using:

\[
P(n) = \{ U \in S(n) | U > 0 \}, \tag{2}
\]

where \( U > 0 \) means that the quadratic form \( x^T U x > 0 \) in terms of all non-zero \( n \)-dimensional vectors \( x \).

The difference between two positive-definite matrices can be determined using a distance, a divergence, or other measures. It is crucial to accurately specify these metrics.
correctly in the application, because the different metrics on $P(n)$ will lead to its different geometric structures.

2.1. Metrics on $P(n)$

On $P(n)$, some metrics can be used as the measurements. Here, we introduce five specific measures including the Frobenius metric, the affine-invariant Riemannian metric, the log-Euclidean metric, the Kullback–Leibler divergence, and the symmetric Kullback–Leibler divergence.

(1) Euclidean Distance

The Frobenius metric (FM) on $P(n)$ is indicated by

$$\langle U, W \rangle_F = \text{tr}(U^T W),$$

where $\text{tr}(\cdot)$ refers to the trace of “$\cdot$” and $U, W \in P(n)$. Then, $P(n)$ with the metric (3) becomes a manifold. The tangent space at the point $U \in P(n)$ can be represented by $T_U P(n)$. Also, the tangent space $T_U P(n)$ at $U$ can be recognized as $S(n)$, as $P(n)$ is an open subset of $S(n)$. The Euclidean distance with the metric (3) corresponds to

$$d_F(U, W) = \|U - W\|_F.$$

(2) Riemannian Distance

The manifold $P(n)$ is transformed into a Riemannian manifold, which is equipped with the affine-invariant Riemannian metric (AIRM) at point $U$

$$g_U(A, B) = g_I(U^{-1} A U^{-1} B) = \text{tr}(U^{-1} A U^{-1} B),$$

where $I$ represents the identity matrix, and $A, B \in T_U P(n)$. With the metric (5), the curvature of $P(n)$ is non-positive [11,12]. The distance of two points $U, W \in P(n)$ induced by the AIRM is provided by the length of the local geodesic:

$$d_{\text{AIRM}}(U, W) = \| \log(U^{-\frac{1}{2}} W U^{-\frac{1}{2}}) \|_F.$$

However, the calculation cost of the AIRM distance is usually time-consuming in practical applications. An alternative option is the log-Euclidean metric (LEM), which can be indicated as:

$$g_U(A, B) = g_I(D_U \log A, D_U \log B),$$

where $A, B \in T_U P(n)$ and $D_U$ indicate the differential map at the point $U$. The LEM distance between $U$ and $W$ could be written as

$$d_{\text{LEM}}(U, W) = \| \log U - \log W \|_F,$$

which is obviously more concise than the AIRM distance.

(3) Divergences

Other geometric measures can be equipped on the manifold $P(n)$ [13]. One widely used measure on Riemannian manifolds is the Kullback–Leibler divergence (KLD), also known as Stein loss or log-determinant divergence [14]. The KLD between two points $U$ and $W$ can be indicated as:

$$d_{\text{KLD}}(U, W) = \text{tr}(W^{-1} I - \log W) - \log \det(W^{-1} U).$$

The symmetric Kullback–Leibler divergence (SKLD) is the Jeffreys divergence [15], and the divergence between $U$ and $W$ is represented by

$$d_{\text{SKLD}}(U, W) = \frac{1}{2} (d_{\text{KLD}}(U, W) + d_{\text{KLD}}(U, W)) = \frac{1}{2} \text{tr}(W^{-1} U + U^{-1} W - 2I).$$
When dealing with optimization problems on manifolds, it is often necessary to calculate the gradient of the objective function \( F(U) \), which can be achieved using the covariant/directional derivative correlated with a given metric, such as a Riemannian metric etc:

\[
\langle \nabla F(U), A \rangle = \frac{d}{d\tau} \bigg|_{\tau=0} F(\gamma(\tau)), \quad \forall A \in T_UP(n),
\]

with the curve \( \gamma : [0,1] \to P(n) \) and \( \gamma(0) = U, \dot{\gamma}(0) = A \). The curve \( \gamma \) is linearized and can be re-expressed by

\[
\langle \nabla F(U), A \rangle = \frac{d}{d\tau} \bigg|_{\tau=0} F(U + \tau A)), \quad \forall A \in T_UP(n).
\]

2.2. Geometric Means on \( P(n) \)

The arithmetic mean of \( m \) positive real numbers \( \{u_i\} \) can be obtained by

\[
\hat{u} := \frac{1}{m} \sum_{i=1}^{m} u_i = \arg \min_{u > 0} \sum_{i=1}^{m} |u - u_i|^2,
\]

where \( \arg \min_{u > 0} \) represents the value of \( u \) that minimizes \( \sum_{i=1}^{m} |u - u_i|^2 \) within the range of \( u > 0 \). For a given set of \( m \) matrices \( U_i \in P(n) \), the mean can be obtained by solving the following minimum problem:

\[
\hat{U} := \arg \min_{U \in P(n)} \sum_{i=1}^{m} d^2(U, U_i).
\]

The arithmetic mean, which is triggered by the Frobenius metric (3), is represented as

\[
\hat{U} = \frac{1}{m} \sum_{i=1}^{m} U_i.
\]

It is shown that the geometric mean of a set of matrices on \( P(n) \) may not have an explicit expression, which can be numerically calculated by a fixed point iterative algorithm \([16–18]\). Table 1 presents the algorithms used to calculate the geometric means.

Table 1. Geometric means corresponding to different metrics.

<table>
<thead>
<tr>
<th>Geometric Metric</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIRM</td>
<td>( \hat{U}_{t+1} = (\hat{U}<em>t)^{\frac{1}{2}} \exp(-\eta \sum</em>{i=1}^{m} \log((\hat{U}_i)^{-\frac{1}{2}}U_i(\hat{U}_i)^{-\frac{1}{2}}))(\hat{U}_t)^{\frac{1}{2}} )</td>
</tr>
<tr>
<td>LEM</td>
<td>( \hat{U} = \exp\left(\frac{1}{m} \sum_{i=1}^{m} \log U_i\right) )</td>
</tr>
<tr>
<td>KLD</td>
<td>( \hat{U} = (\frac{1}{m} \sum_{i=1}^{m} U_i)^{-1} )</td>
</tr>
<tr>
<td>SKLD</td>
<td>( \hat{U} = ((\frac{1}{m} \sum_{i=1}^{m} U_i)^{-1})^{-1} )</td>
</tr>
</tbody>
</table>

3. Point Cloud Denoising Algorithm Based on Geometric Metrics

Let us represent the point cloud with the scale \( \kappa \) as

\[
\mathcal{D} := \{c \in \mathbb{R}^n \mid j = 1, \ldots, \kappa\}.
\]

In terms of any \( c \in \mathcal{D} \), the \( q \)-nearest neighbor method can be adopted for selecting the neighborhood \( N(c, q) \), with the abbreviation of \( \mathcal{N} \). Next, we calculate the expectation \( \mu(\mathcal{N}) := E(\mathcal{N} - c) \) and covariance matrix \( \Sigma(\mathcal{N}) := \text{Cov}(\mathcal{N}) \) of the data points in \( \mathcal{N} \), respec-
tively. Therefore, we map original data point cloud onto the Gaussian distribution family manifold, aiming to acquire the parameter point cloud \( N_n \) as follows
\[
\mathcal{N}_n = \{ P | P = P(x, \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\{-\frac{(x - \mu)^T\Sigma^{-1}(x - \mu)}{2}\} \}.
\]
Then, the local statistical mapping is represented as
\[
\Psi : \mathcal{D} \rightarrow \mathcal{N}_n,
\]
and satisfies
\[
\Psi(P) = P(\mu(\mathcal{N}_n), \Sigma(\mathcal{N}_n)) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\{-\frac{(x - \mu)^T\Sigma^{-1}(x - \mu)}{2}\}.
\]

Based on the local statistical mapping \( \Psi \), the image \( \tilde{\mathcal{D}} := \Psi(\mathcal{D}) \subseteq \mathcal{N}_n \) of the point cloud \( \mathcal{D} \) refers to the parameter point cloud \( \mathcal{N}_n \).

3.1. K-Means Clustering Algorithm on the Basis of Geometric Metrics

Since \( \mathcal{N}_n \) and \( \mathbb{R}^n \times \mathcal{P}(n) \) are shown to be topologically homeomorphic \[19\], the geometric structure on \( \mathcal{N}_n \) can be triggered by setting the metrics on \( \mathbb{R}^n \times \mathcal{P}(n) \). In addition, we denote the distance on \( \mathcal{N}_n \) as:
\[
d((\mu_1, \Sigma_1), (\mu_2, \Sigma_2)) = \|\mu_1 - \mu_2\|_F + d(\Sigma_1, \Sigma_2),
\]
and the barycenter of the parameter point cloud \( \tilde{\mathcal{D}} \) is indicated:
\[
g(\tilde{\mathcal{D}}) = \arg \min_{(\mu, \Sigma)} \sum_{i=1}^{k} d^2((\mu_i, \Sigma_i), (\mu, \Sigma)) = (\frac{1}{k} \sum_{i=1}^{k} \mu_i, \arg \min_{\Sigma} \sum_{i=1}^{k} d^2(\Sigma_i, \Sigma)),
\]
where the barycenter \( g(\tilde{\mathcal{D}}) \) represents the geometric center of \( \tilde{\mathcal{D}} \). The distance and barycenter \( g(\tilde{\mathcal{D}}) \) rely on the selection of the metric on \( \mathcal{N}_n \). The distance functions in the following algorithm can be induced by FM, AIRM, LEM, KLD, or SKLD, and the geometric means corresponding to each metric can be found in Table 1.

The local statistical structure of the valid data and that of random noise are very different, so we use the K-means clustering algorithm to divide \( \tilde{\mathcal{D}} \) into two categories. The algorithm for clustering data and noise is presented below:

**Algorithm 1 Algorithm for clustering signal and noise**

1. Mapping the original point cloud \( \mathcal{D} \) to a parameter point cloud \( \tilde{\mathcal{D}} \).
2. K-means algorithm:
   (a) Set the barycenters of the current division as \( g_1^{(i)} \) and \( g_2^{(i)} \), and apply the K-means algorithm based on the distance function (4), (6), (8), (9) or (10) to group the parameter point cloud \( \tilde{\mathcal{D}} \).
   (b) Update the barycenters of the current division to \( g_1^{(i+1)} \) and \( g_2^{(i+1)} \) based on the clustering results, where the geometric mean of each category is calculated by Table 1.
   (c) Set the threshold \( \epsilon_0 \). When \( d(g_1^{(i)}, g_1^{(i+1)}) < \epsilon_0 \) and \( d(g_2^{(i)}, g_2^{(i+1)}) < \epsilon_0 \), the current division of \( \tilde{\mathcal{D}} \) is mapped to two categories of the original point cloud, and then the program ends. Otherwise, run Step (b).
3.2. Influence Functions of Different Metrics

In order to analyze the robustness of geometric means when symmetry positive-definite matrices are contaminated by outliers, we adopt the influence function. To show that it is more suitable to assign geometric metrics than the Euclidean metric in Algorithm 1, we will describe the influence functions of the arithmetic mean and the geometric means.

Let $\overline{U}$ indicate the mean of $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ with FM (AIRM, LEM, KLD, or SKLD), and let $\hat{U}$ be the mean by supplementing a set of $l$ outliers $R_1, R_2, \ldots, R_l$ with a weight $\tau (\tau \ll 1)$ to $U_1, U_2, \ldots, U_m$ [20,21].

Next, we can define $\hat{U}$ as:

$$\hat{U} = \overline{U} + \tau H(\hat{U}, R_1, R_2, \ldots, R_l) + O(\tau^2)$$

and the norm as

$$h(\hat{U}, R_1, R_2, \ldots, R_l) := \|H(\hat{U}, R_1, R_2, \ldots, R_l)\|_F.$$  

The influence functions of the arithmetic mean and the AIRM mean can be found in Refs. [10,22], as follows.

**Proposition 1.** The influence function of the arithmetic mean for $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ and $l$ outliers $R_1, R_2, \ldots, R_l$ with a weight $\tau (\tau \ll 1)$ can be expressed as:

$$H(\hat{U}, R_1, R_2, \ldots, R_l) = \frac{1}{l} \sum_{j=1}^{l} R_j - \overline{U}.$$  

**Proposition 2.** The influence function of the AIRM mean for $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ and $l$ outliers $R_1, R_2, \ldots, R_l$ is given by

$$H(\hat{U}, R_1, R_2, \ldots, R_l) = -\frac{1}{l} \sum_{j=1}^{l} \overline{U} \log(R_j^{-1} \overline{U}).$$  

Before the influence functions of the LEM mean, the KLD mean, as well as the SKLD mean are presented, the following lemmas are given to calculate them [23,24].
Lemma 1. Let $V$ and $Z(\tau), \forall t \in \mathbb{R}$ be an invertible matrix that does not have eigenvalues in the closed negative real line, and let $\log V$ be its principal logarithm, then we have that
1. Both $V$ and $\log V$ commute with $[(V - I)t + I]^{-1}$ for any $t \in \mathbb{R}$.
2. The following identities are valid that
   \begin{align*}
   \int_0^1 [(V - I)t + I]^{-2} dt &= (I - V)^{-1}[(V - I)t + I]^{-1}\big|_{t=0} = V^{-1} \\
   \frac{d}{d\tau} \log(Z(\tau)) &= \int_0^1 [(Z(\tau) - I)t + I]^{-1} \frac{dZ(\tau)}{d\tau} [(Z(\tau) - I)t + I]^{-1} dt.
   \end{align*}

Lemma 2. Suppose that $Z(t)$ is a real matrix for $\forall t \in \mathbb{R}$, then we have
\begin{equation}
\text{tr}\left(\int_a^b Z(t) dt\right) = \int_a^b \text{tr}(Z(t)) dt.
\end{equation}

Lemma 3. For any invertible matrix $Z(t)$, the following identity holds
\begin{equation}
\frac{d}{d\tau} Z(t) = |Z(t)| \text{tr}(Z^{-1} \frac{d}{d\tau} Z(t)).
\end{equation}

Then, the influence functions of the LEM mean, the KLD mean, and the SKLD mean are presented as:

Proposition 3. The influence function of the LEM mean for $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ and $l$ outliers $R_1, R_2, \ldots, R_l$ is represented by
\begin{equation}
H(\hat{U}, R_1, R_2, \ldots, R_l) = \frac{1}{l} \sum_{j=1}^l (\log R_j - \log \hat{U}).
\end{equation}

Proof. See Appendix A. \qed

Proposition 4. The influence function of the KLD mean for $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ and $l$ outliers $R_1, R_2, \ldots, R_l$ is represented by
\begin{equation}
H(\hat{U}, R_1, R_2, \ldots, R_l) = \hat{U} - \frac{1}{l} \sum_{j=1}^l U_j R_j^{-1} \hat{U}.
\end{equation}

Proof. See Appendix B. \qed

Proposition 5. The influence function of the SKLD mean for $m$ symmetry positive-definite matrices $U_1, U_2, \ldots, U_m$ and $l$ outliers $R_1, R_2, \ldots, R_l$ is represented by
\begin{equation}
H(\hat{U}, R_1, R_2, \ldots, R_l) = \frac{1}{2l} \sum_{j=1}^l (R_j - \hat{U} R_j^{-1} \hat{U}).
\end{equation}

Proof. See Appendix C. \qed

4. Simulations and Results

The following section presents numerical experiments to demonstrate the denoising effect of Algorithm 1. It also compares the norms of the mean influence functions with FM, AIRM, LEM, KLD, and SKLD. In these simulations, all samples on $P(n)$ can be generated using the equation:
\begin{equation}
\exp\left(\frac{V + V^T}{2}\right),
\end{equation}
where \( V \in \mathbb{R}^{n \times n} \) refers to a random matrix generated by MATLAB.

A. Numerical Simulations

In this example, the distance function induced by FM, AIRM, LEM, KLD, or SKLD is used in Algorithm 1, and the denoising effect caused by these metrics is compared, where the experimental data employ the 3D point cloud of Teapot.ply. Teapot.ply is one of the built-in data in MATLAB and can also be obtained through some open-source projects, such as the Stanford 3D Scanning Repository. In MATLAB, Teapot.ply is an example model that is often used to demonstrate and test 3D graphics processing and visualization functions. It is stored in PLY file format and contains three-dimensional geometric information representing a teapot.

As shown in Figure 2a, background noise can be distributed uniformly with a signal-to-noise ratio (SNR) of 4137:1000. Taking \( K = 2, q = 5 \) and \( \epsilon_0 = 0.01 \), Figure 2b–f present the denoised images of the original point cloud using Algorithm 1, according to FM, AIRM, LEM, KLD, and SKLD, respectively. From Figure 2b, it can be observed that Algorithm 1, which is based on the Euclidean metric, still has a significant amount of noise points remaining in the image. Figure 2c,d show that Algorithm 1 performs well in removing noise points when using AIRM and LEM. Figure 2e,f illustrate that Algorithm 1, utilizing KLD or SKLD, effectively denoises the image but also removes some real data points from the teapot. It is evident that Algorithm 1 on the basis of the Euclidean metric is not as effective as other geometric metrics.

![Figure 2. Comparison of point cloud before and after denoising.](image-url)
B. Results and Discussions

To evaluate the denoising effects of these metrics, true positive (TP), false positive (FP), false negative (FN), and true negative (TN) are adopted. Then, the true positive rate (TPR), the false positive rate (FPR), and the signal–noise rate growing (SNRG) are defined by

\[
\text{TPR} = \frac{TP}{N_{\text{data}}}, \quad \text{FPR} = \frac{FP}{N_{\text{noise}}}, \quad \text{SNRG} = \frac{TP}{FP} \left( \frac{N_{\text{noise}}}{N_{\text{data}}} - 1 \right),
\]

where \(N_{\text{data}}\) represents the number of real data, and \(N_{\text{noise}}\) stands for the number of noises.

In Table 2, we have an SNR of 4137:1000 and 4137:2000. As higher TPR, higher SNRG, and lower FPR indicate that the algorithm can better distinguish between real data and noise, the highest TPR, the lowest FPR, and more than 99% SNRG are displayed in bold. From Table 2, it is evident that Algorithm 1 based on the Euclidean metric generally has a higher FPR, lower TPR, and lower SNRG. Conversely, geometric metrics tend to lower FPR, higher TPR, and higher SNRG. Among them, Algorithm 1 with KLD and SKLD exhibits a lower FPR due to the removal of some valid data points in Figure 2e,f. Therefore, Table 2 demonstrates the advantages of using geometric metrics (AIRM, LEM, KLD, and SKLD) over the Euclidean metric.

### Table 2. Comparison of denoising results.

<table>
<thead>
<tr>
<th>Metric</th>
<th>SNR 4137:1000</th>
<th>SNR 4137:2000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TPR</td>
<td>FPR</td>
</tr>
<tr>
<td>Arith</td>
<td>99.59%</td>
<td>74.30%</td>
</tr>
<tr>
<td>AIRM</td>
<td>100.00%</td>
<td>42.50%</td>
</tr>
<tr>
<td>LEM</td>
<td>100.00%</td>
<td>42.50%</td>
</tr>
<tr>
<td>KLD</td>
<td>77.45%</td>
<td>34.90%</td>
</tr>
<tr>
<td>SKLD</td>
<td>94.95%</td>
<td>34.00%</td>
</tr>
</tbody>
</table>

To compare the robustness of influence functions with different metrics, we consider simulations involving 100 randomly generated symmetry positive-definite matrices along with \(l\)-injected outliers. We examine four scenarios: \(l = 10, 40, 70,\) and \(100\) to analyze the effect of outlier numbers on influence function robustness.

From Propositions 1–5 as a basis for calculation norms of influence functions, we repeat simulations for each scenario (\(l = 10, 40, 70,\) or \(100\)) one hundred times, as shown in Figure 3. It can be seen that the norms of the influence functions corresponding to the geometric metrics (AIRM, LEM, KLD, and SKLD) are not sensitive to changes in outlier numbers; they remain within a range close to one, regardless of if \(l\) is set from 10 up to 100. On the other hand, norms associated with Euclidean metric (FM) fluctuate significantly—decreasing from around seven or eight down towards one when \(l\) increases from 10 up to 100. Therefore, geometric means are almost independent of the number of outliers, and more stable than the arithmetic mean.

Figure 4 provides an intuitive representation by illustrating average norms across all simulations for each metric type (Euclidean vs Geometric). As observed in Figure 3, results hold true here as well: FM consistently yields larger norm values compared to those obtained using geometric metrics when considering various values for \(l\) such as 10, 40, and 70, respectively. Furthermore, Figure 4 shows that if the number of outliers with the FM is 100, which is close to the number of symmetry positive-definite matrices used in simulations, influence functions for FM and geometric metrics become increasingly similar. This further supports the notion that geometric means are more robust than arithmetic means, as demonstrated by Figure 4.
Figure 3. Norms of influence functions.

Figure 4. Means of influence functions.
C. Complexity Analysis

Next, the computational complexity of the mean matrices with FM, LEM, AIRM, KLD, and SKLD will be demonstrated, whose formulas can be obtained from (15) and Table 1. Then, the computational complexity of the influence functions according to FM, LEM, AIRM, KLD, and SKLD will also be given, whose expressions are shown in Propositions 1–5. For iterative algorithms, we only calculate the complexity of single-step iterations.

This subsection demonstrates the computational complexity of mean matrices for FM, LEM, AIRM, KLD, and SKLD. The formulas can be obtained from (15) and Table 1. Additionally, the computational complexity of influence functions according to these metrics is provided in Properties 1–5. For iterative algorithms, only the complexity of single-step iterations is considered. Let us assume we have \( m \times n \) symmetry positive-definite matrices and \( l \) outliers. It is assumed that the complexity of a single element is \( O(1) \), \( U^{-1} \sim O(n^3) \) and \( \log U \sim O(n^4) \). When calculating the matrix exponential of a symmetric matrix and the half power of a symmetric positive-definite matrix, the eigenvalue decomposition is needed to perform. Thus, we have that \( \exp(A) \sim O(n^3) \) and \( U^{1/2} \sim O(n^3) \).

Table 3 shows that the arithmetic mean has the lowest computation time followed by KLD and SKLD means. As the complexity of each step of the AIRM mean is equivalent to that of the LEM mean, so the LEM mean has a much faster calculation speed than the AIRM mean.

From Table 4, it can be seen that calculating the influence functions with the FM takes less time than those with the geometric metrics. Among geometric metrics, the longest calculation time is the influence function corresponding to the AIRM, followed by that corresponding to the LEM, and the shortest calculation time is the influence function induced by the KLD and the SKLD. This difference arises due to the iterative algorithm requirement for calculating AIRM.

Table 3. Computational complexity of means.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM</td>
<td>( O((m-1)n^2) )</td>
</tr>
<tr>
<td>AIRM</td>
<td>( O((m-1)n^4) )</td>
</tr>
<tr>
<td>LEM</td>
<td>( O(mn^4) )</td>
</tr>
<tr>
<td>KLD</td>
<td>( O((m+1)n^3) )</td>
</tr>
<tr>
<td>SKLD</td>
<td>( O((m+2)n^3) )</td>
</tr>
</tbody>
</table>

Table 4. Computational complexity of influence functions.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM (Proposition 1)</td>
<td>( O((m+l-2)n^2) )</td>
</tr>
<tr>
<td>AIRM (Proposition 2)</td>
<td>( O((2m+1)n^4) )</td>
</tr>
<tr>
<td>LEM (Proposition 3)</td>
<td>( O((2m+1)n^4) )</td>
</tr>
<tr>
<td>KLD (Proposition 4)</td>
<td>( O((2m+3)(l+m+1)n^3) )</td>
</tr>
<tr>
<td>SKLD (Proposition 5)</td>
<td>( O((2m+3)n^3) )</td>
</tr>
</tbody>
</table>

5. Conclusions

To conclude, a novel point cloud denoising algorithm is proposed in combination with the K-means algorithm. By calculating the expectation and covariance of the data points, this algorithm utilizes geometric metrics on the statistical manifold to map the original data point cloud to the Gaussian distribution family manifold, forming a parameter point cloud. Additionally, different measure structures are constructed on the Gaussian distribution family manifold, and the K-means method is used for clustering the parameter point cloud, aiming to cluster the corresponding original data. The robustness of different means with various metrics is analyzed by calculating their approximate influence functions. Simulations show that Algorithm 1 with the geometry metrics can obtain a better denoising
effect than that with the Euclidean metric, and the geometric means are more robust than the arithmetic mean. We use three criteria (TPR, FPR, and SNRG) to evaluate the denoising effect of Algorithm 1 corresponding to different metrics. Simulations indicate that the algorithm corresponding to the Euclidean metric has a lower TPR and a higher FPR in most cases. SNRG is the most important criteria, which represents the promotion of signal-to-noise ratio through Algorithm 1. Table 2 shows that the SNRG of Algorithm 1 with geometric metrics is generally higher than that with the Euclidean metric.

However, it should be noted that although denoising algorithms based on geometric metrics have the advantages of high TPR, low FPR, and high SNRG, it faces higher computational complexity than those utilizing the Euclidean metric, reaching $O(n^4)$. Additionally, computational complexity according to KLD and SKLD also reaches $O(n^3)$. This represents a drawback of denoising algorithms employing geometric metrics.

In this manuscript, Algorithm 1 is only used for numerical simulation of point cloud denoising, and can also be applied to specific tests in the future. Ultrasonic testing with a manipulator is an important non-destructive testing method used to detect the internal defects of materials with complex structures. Before the manipulator detects the work-piece, the laser instrument is used to collect the point cloud data of the workpiece for 3D reconstruction. However, the point cloud data collected by laser instruments often contains a significant amount of redundant or invalid points. Algorithm 1 can be used to these redundancies and improve the quality of point cloud data. This will result in reduced noise and more accurate models, ultimately enhancing subsequent processing effects.

Author Contributions: Investigation, X.D.; Methodology, X.D. and L.F.; Software, X.Z.; Writing—original draft, X.D.; Writing—review and editing, X.D., L.F. and X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China grant number 61401058.

Data Availability Statement: The experimental data employ 3D point cloud of Teapot.ply. Teapot.ply is one of the built-in data in MATLAB and can also be obtained through some open source projects, such as the Stanford 3D Scanning Repository.

Acknowledgments: The authors would like to thank the anonymous reviewers for their detailed and careful comments, which helped improve the quality of presentation.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Proof of Proposition 3. Let $F(U)$ represent the objective function to minimize $m$ symmetry positive-definite matrices as follows:

$$F(U) := \frac{1}{m} \sum_{i=1}^{m} \| \log U - \log U_i \|^2_F.$$  \hspace{1cm} (A1)

Suppose $Z(\tau) := U + \tau A$, we can choose an appropriate $\tau > 0$ so that $Z(\tau)$ is positive-definite. In fact, $A$ belongs to the tangent vector space $T_U PD(n)$, and the geodesic $\gamma$ passing through $U$ in the direction of $A$ with respect to the Frobenius inner product is given by

$$\gamma(\tau) = U + \tau A.$$  \hspace{1cm} (A2)

According to Proposition 2.2.5 of [25], $\gamma(\tau)$ stays within $PD(n)$ under the condition that $\tau$ is not too large. Therefore, we may select the appropriate $\tau$ to guarantee $Z(\tau) \in PD(n)$. Then, the gradient of $F(U)$ can be computed as follows by (12)
\[ \langle \nabla F(U), A \rangle = \left. \frac{d}{d\tau} \right|_{\tau=0} F(U + \tau A) \]
\[ = \frac{1}{m} \frac{d}{d\tau} \left|_{\tau=0} \text{tr} \left( \sum_{i=1}^{m} (\log Z(\tau) - \log U_i)^2 \right) \right. \]
\[ = \frac{2}{m} \sum_{i=1}^{m} \text{tr} \left( \left( \log Z(\tau) - \log U_i \right) \frac{d}{d\tau} \log Z(\tau) \right) \]
\[ = 2 \text{tr} \left( \log Z(\tau) \frac{d}{d\tau} \log Z(\tau) \right) \left|_{\tau=0} \right. - \frac{2}{m} \sum_{i=1}^{m} \text{tr} \left( \log U_i \frac{d}{d\tau} \log Z(\tau) \right) \left|_{\tau=0} \right. . \]

The rest of the analysis is to calculate the two differential terms of the above identity. Using Lemmas 1–3, we obtain
\[
\text{tr} \left( \log Z(\tau) \frac{d}{d\tau} \log Z(\tau) \right) = \text{tr} \left( \int_{0}^{1} \log Z(\tau)(Z(\tau) - I)t + I]^{-1} dZ(\tau) \right) \]
\[ = \text{tr} \left( \int_{0}^{1} [Z(\tau) - I]t + I]^{-1} \log Z(\tau) \frac{dZ(\tau)}{d\tau} [Z(\tau) - I]t + I]^{-1} dt \right) \]
\[ = \text{tr} \left( \int_{0}^{1} [Z(\tau) - I]t + I]^{-1} \log Z(\tau) \frac{dZ(\tau)}{d\tau} [Z(\tau) - I]t + I]^{-1} dt \right) \]
\[ = \text{tr} \left( \int_{0}^{1} [Z(\tau) - I]t + I]^{-2} \log Z(\tau) \frac{dZ(\tau)}{d\tau} \right) \]
\[ = \text{tr} \left( \int_{0}^{1} [Z(\tau) - I]t + I]^{-2} \log Z(\tau) \frac{dZ(\tau)}{d\tau} \right) \]
\[ = \text{tr} \left( Z(\tau)^{-1} \log Z(\tau) A \right) \]

and
\[
\text{tr} \left( \log U_i \frac{d}{d\tau} \log Z(\tau) \right) = \text{tr} \left( Z(\tau)^{-1} \log(U_i) A \right) \]. \quad (A3)

Then, we obtain
\[
\langle \nabla F(U), A \rangle = \text{tr} \left( \frac{2}{m} \sum_{i=1}^{m} U^{-1} (\log U - \log U_i) A \right) \]
\[ = \text{tr} \left( \frac{2}{m} \sum_{i=1}^{m} U^{-1} (\log U - \log U_i) A \right) \] \quad (A4)

and thus
\[
\nabla F(U) = \frac{2}{m} \sum_{i=1}^{m} (\log U - \log U_i) U^{-1}. \quad (A5)
\]

Let \( G(U) \) be the objective function, aiming to minimize the \( m + l \) symmetry positive-definite matrices as follows
\[
G(U) := (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} ||\log U - \log U_i||^2 + \tau \frac{1}{l} \sum_{j=1}^{l} ||\log U - \log R_j||^2, \]
\[ = (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} ||\log U - \log U_i||^2 + \tau \frac{1}{l} \sum_{j=1}^{l} ||\log U - \log R_j||^2, \quad (A6) \]

then we have
\[
\nabla G(U) = (1 - \tau) \frac{2}{m} \sum_{i=1}^{m} (\log U - \log U_i) U^{-1} + \tau \frac{2}{l} \sum_{j=1}^{l} (\log U - \log R_j) U^{-1}. \quad (A7)
\]

Note that \( \hat{U} = \bar{U} + \tau H(\hat{U}, R_1, R_2, \ldots, R_l) + O(\tau^2) \) is the mean of \( m \) symmetry positive-definite matrices \( U_1, U_2, \ldots, U_m \) and \( l \) outliers \( R_1, R_2, \ldots, R_l \), so we obtain \( \nabla G(\hat{U}) = 0 \), that is,
\[(1 - \tau) \frac{1}{m} \sum_{i=1}^{m} (\log \hat{U} - \log U_i) + \tau \frac{1}{l} \sum_{j=1}^{l} (\log \hat{U} - \log R_j) = 0. \quad (A8)\]

with the purpose of obtaining the linear term about \( \tau \), this study differentiates \((A8)\) and sets \( \tau = 0 \). Thus, we obtain

\[
\frac{1}{l} \sum_{j=1}^{l} (\log \overline{U} - \log R_j) + \frac{1}{m} \sum_{i=1}^{m} \frac{d}{d\tau} \bigg|_{\tau=0} (\log \hat{U} - \log U_i) = 0. \quad (A9)\]

As \( \overline{U} \) represents the mean of \( m \) symmetry positive-definite matrices \( U_1, U_2, \ldots, U_m \), we obtain that

\[
\frac{1}{m} \sum_{i=1}^{m} (\log \overline{U} - \log U_i) = 0. \quad (A10)\]

Next, we take the trace of Equation \((A9)\) and apply Lemmas 1–3 to obtain

\[
\text{tr} \left( \log \overline{U} - \frac{1}{l} \sum_{j=1}^{l} \log R_j + \overline{U}^{-1} H \right) = 0. \quad (A11)\]

Based on the arbitrariness of \( \overline{U} \), this study can obtain the influence function of the LEM mean that

\[
H = \frac{1}{l} \sum_{j=1}^{l} \overline{U} (\log R_j - \log \overline{U}). \quad (A12)\]

This can finish the proof. \( \square \)

Appendix B

**Proof of Proposition 4.** Let \( F(U) \) be the objective function in order to minimize \( m \) positive-definite matrices as follows

\[
F(U) := \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( U_i^{-1}U - \log(U_i^{-1}U) - I \right). \quad (A13)\]

Suppose \( Z(\tau) := U + \tau A \), then the gradient of \( F(U) \) can be computed as follows by \((12)\)

\[
\langle \nabla F(U), A \rangle = \frac{d}{d\tau} \bigg|_{\tau=0} F(U + \tau A) = \frac{d}{d\tau} \bigg|_{\tau=0} \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( U_i^{-1}Z(\tau) - \log(U_i^{-1}Z(\tau)) - I \right) = \frac{1}{m} \sum_{i=1}^{m} \text{tr} (U_i^{-1}A) - \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( \frac{d}{d\tau} \bigg|_{\tau=0} \log(U_i^{-1}U) \right).\]

The rest is to calculate the differential term of the above identity. Using Lemmas 1–3, we obtain

\[
\text{tr} (\log Z(\tau) \frac{d}{d\tau} \log(U_i^{-1}Z(\tau))) = \int_0^1 \text{tr} \left( [(U_i^{-1}Z(\tau) - I)t + I]^{-1} \frac{d(U_i^{-1}Z(\tau))}{d\tau} [(U_i^{-1}Z(\tau) - I)t + I]^{-1} \right) dt = \int_0^1 \text{tr} \left( [(U_i^{-1}Z(\tau) - I)t + I]^{-2} dt U_i^{-1}A \right) = \text{tr} \left( (U_i^{-1}U)^{-1}U_i^{-1}A \right) = \text{tr} \left( U^{-1}A \right).\]
Consequently, we obtain
\[
\nabla F(U) = \text{tr} \left( \frac{1}{m} \sum_{i=1}^{m} (U^{-1}_i - \overline{U}) \right). \quad (A14)
\]

Let \( G(U) \) be the objective function to minimize the \( m + l \) symmetry positive-definite matrices as follows
\[
G(U) := (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( U^{-1}_i U - \log(U^{-1}_i U) - I \right) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr} \left( R^{-1}_j U - \log(R^{-1}_j U) - I \right),
\]
then we have
\[
\nabla G(U) = (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr} (U^{-1}_i - U^{-1}) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr} (R^{-1}_j - U^{-1}). \quad (A15)
\]
As \( \nabla G(\overline{U}) = 0 \), we have
\[
(1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr} (U^{-1}_i - \overline{U}) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr} (R^{-1}_j - \overline{U}) = 0. \quad (A16)
\]
with the aim of obtaining the linear term about \( \tau \), this study differentiates (A16) and sets \( \tau = 0 \). Thus, we obtain
\[
\frac{1}{l} \sum_{j=1}^{l} \text{tr} (R^{-1}_j - U^{-1}) + \frac{1}{m} \sum_{i=1}^{m} \frac{d}{d\tau} \bigg|_{\tau=0} \text{tr} (U^{-1}_i - U^{-1}) = 0. \quad (A17)
\]
Note that
\[
0 = \frac{d}{d\tau} I = \frac{d}{d\tau} (\overline{U} \overline{U}^{-1}) = \left( \frac{d}{d\tau} \overline{U} \right) \overline{U}^{-1} + \overline{U} \frac{d}{d\tau} \overline{U}^{-1}, \quad (A18)
\]
consequently,
\[
\frac{d}{d\tau} \bigg|_{\tau=0} \overline{U}^{-1} = -\overline{U}^{-1} \frac{d}{d\tau} \overline{U} \bigg|_{\tau=0} = -\overline{U}^{-1} H \overline{U}^{-1}. \quad (A19)
\]
Next, we substitute (A19) into (A17) and take the trace of Equation (A17) to obtain the influence function of the KLD mean:
\[
H = \overline{U} - \frac{1}{l} \sum_{j=1}^{l} \overline{U} R^{-1}_j \overline{U}. \quad (A20)
\]

Appendix C

Proof of Proposition 5. Let \( F(U) \) be the objective function to minimize \( m \) symmetry positive-definite matrices as follows
\[
F(U) := \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( U^{-1}_i U + U^{-1} U_i - 2I \right). \quad (A21)
\]
According to (12), the gradient of \( F(U) \) is computed by:
with $Z(\tau) := U + \tau A$. Using (A18), we obtain
\[
\text{tr}\left( \frac{d}{d\tau} \right)_{\tau=0} Z(\tau)^{-1} U_l = -\text{tr}\left( U^{-1} U_l U^{-1} A \right). \tag{A22}
\]
Consequently, we obtain the gradient of $F(U)$ that
\[
\nabla F(U) = \frac{1}{m} \sum_{i=1}^{m} \left( U_i^{-1} - U^{-1} U_l U^{-1} \right). \tag{A23}
\]
Let $G(U)$ be the objective function to minimize $m + l$ symmetry positive-definite matrices as follows
\[
G(U) := (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr} \left( U_i^{-1} U + U_i^{-1} U_l - 2I \right) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr} \left( R_j^{-1} U + U^{-1} R_j - 2I \right),
\]
then we have
\[
\nabla G(U) = (1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr}(U_i^{-1} - U^{-1} U_l U^{-1}) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr}(R_j^{-1} - U^{-1} R_j U^{-1}). \tag{A24}
\]
As $\nabla G(\hat{U}) = 0$, we have
\[
(1 - \tau) \frac{1}{m} \sum_{i=1}^{m} \text{tr}(U_i^{-1} - \hat{U}_i^{-1} \hat{U}_l \hat{U}_i^{-1}) + \tau \frac{1}{l} \sum_{j=1}^{l} \text{tr}(R_j^{-1} - \hat{U}_j^{-1} \hat{R}_j \hat{U}_j^{-1}) = 0. \tag{A25}
\]
In order to acquire the linear term concerning $\tau$, this study differentiates (A25) and sets $\tau = 0$, namely,
\[
\frac{1}{l} \sum_{j=1}^{l} \text{tr}(R_j^{-1} - \hat{U}_j^{-1} \hat{R}_j \hat{U}_j^{-1}) + \frac{1}{m} \sum_{i=1}^{m} \frac{d}{d\tau} \bigg|_{\tau=0} \text{tr}(U_i^{-1} - \hat{U}_i^{-1} \hat{U}_l \hat{U}_i^{-1}) = 0. \tag{A26}
\]
Note that
\[
\frac{d}{d\tau} \bigg|_{\tau=0} (\hat{U}_l^{-1} \hat{U}_i^{-1}) = -\hat{U}_l^{-1} H \hat{U}_i^{-1} \hat{U}_l^{-1} - \hat{U}_i^{-1} \hat{U}_l^{-1} H \hat{U}_l^{-1}, \tag{A27}
\]
we substitute (A27) into (A26) and take the trace of Equation (A26) to obtain the influence function of the SKLD mean that
\[
H = \frac{1}{2l} \sum_{j=1}^{l} (R_j - \hat{U}_j^{-1} \hat{U}_j). \tag{A28}
\]
References


8. Duan, X.; Ji, X.; Sun, H.; Guo, H. A non-iterative method for the difference of means on the Lie Group of symmetric positive-definite matrices. Mathematics 2022, 10, 255. [CrossRef]


13. Ye, L.; Yang, Q.; Chen, Q.; Deng, W. Multidimensional joint domain localized matrix constant false alarm rate detector based on information geometry method with applications to high frequency surface wave Radar. IEEE Access 2019, 7, 28080–28088. [CrossRef]


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.