Learning manifolds from non-stationary streams

Suchismit Mahapatra[1*] and Varun Chandola[1]

*Correspondence:
suchismi@buffalo.edu

[1] Department of Computer
Science, University at Buffalo,
Buffalo, USA

Abstract

Streaming adaptations of manifold learning based dimensionality reduction methods, such as *Isomap*, are based on the assumption that a small initial batch of observations is enough for exact learning of the manifold, while remaining streaming data instances can be cheaply mapped to this manifold. However, there are no theoretical results to show that this core assumption is valid. Moreover, such methods typically assume that the underlying data distribution is stationary and are not equipped to detect, or handle, sudden changes or gradual drifts in the distribution that may occur when the data is streaming. We present theoretical results to show that the quality of a manifold asymptotically converges as the size of data increases. We then show that a Gaussian Process Regression (GPR) model, that uses a manifold-specific kernel function and is trained on an initial batch of sufficient size, can closely approximate the state-of-art streaming Isomap algorithms, and the predictive variance obtained from the GPR prediction can be employed as an effective detector of changes in the underlying data distribution. Results on several synthetic and real data sets show that the resulting algorithm can effectively learn lower dimensional representation of high dimensional data in a streaming setting, while identifying shifts in the generative distribution. For instance, key findings on a Gas sensor array data set show that our method can detect changes in the underlying data stream, triggered due to real-world factors, such as introduction of a new gas in the system, while efficiently mapping data on a low-dimensional manifold.

Keywords: Manifold learning, Dimension reduction, Streaming data, Isomap, Gaussian process

Mathematics Subject Classification: Primary, 60G15

Introduction

High-dimensional data is inherently difficult to explore and analyze, owing to the "curse of dimensionality" that render many statistical and machine learning techniques inadequate. In this context, *non-linear dimensionality reduction* (NLDR) has proved to be an indispensable tool. Manifold learning based NLDR methods, such as Isomap [1], Local Linear Embedding (LLE) [2], etc., assume that the distribution of the data in the high-dimensional observed space is not uniform. Instead, the data is assumed to lie near a non-linear low-dimensional manifold embedded in the
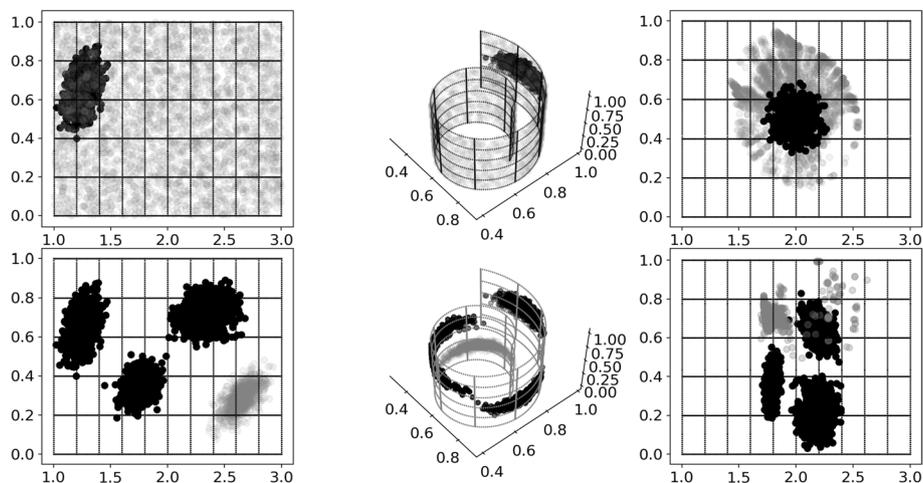
Fig. 1 Impact of changes in the data distribution on streaming NLDR. In the *top* panel, the true data lies on a 2D manifold (*top-left*) and the observed data is in $\mathbb{R}^3$ obtained by using the *swiss-roll* transformation of the 2D data (*top-middle*). The streaming algorithm (S-Isomap [5]) uses a batch of samples from a 2D Gaussian (black), and maps streaming points sampled from a uniform distribution (gray). The streaming algorithm performs well on mapping the batch points to $\mathbb{R}^2$ but fails on the streaming points that "drift" away from the batch (*top-right*). In the *bottom* panel, the streaming algorithm (S-Isomap++ [6]) uses a batch of samples from three 2D Gaussians (black). The stream points are sampled from the three Gaussians and a new Gaussian (gray). The streaming algorithm performs well on mapping the batch points to $\mathbb{R}^2$ but fails on the streaming points that are "shifted" from the batch (*bottom-right*). Both streaming algorithms are discussed in Sect. "Problem statement and preliminaries"

high-dimensional space. By exploiting the geometric properties of the manifold, e.g., smoothness, such methods infer the low-dimensional representation of the data from the high-dimensional observations.

A key shortcoming of NLDR methods is their $O(n^3)$ complexity, where *n* is the size of the data [1]. If directly applied on streaming data, where data arrives one point at a time, NLDR methods have to recompute the entire manifold at every time step, making such a naive adaptation prohibitively expensive. To alleviate the computational problem, landmark-based methods [3] or general out-of-sample extension methods [4] have been proposed. However, these techniques are still computationally expensive for practical applications. Recently, a streaming adaptation of the Isomap algorithm [1], which is a widely used NLDR method, was proposed [5]. This method, called S-Isomap, relies on exact learning from a small initial batch of observations, followed by approximate mapping of subsequent stream of observations. An extension to the case when the observations are sampled from multiple, and possibly intersecting, manifolds, called S-Isomap++, was subsequently proposed [6].

Empirical results on benchmark data sets show that these methods can reliably learn the manifold with a small initial batch of observations. However two issues still remain. *First*, no theoretical bounds on the quality of the manifold, as a function of the initial batch size, exist. *Second*, these methods assume that the underlying generative distribution is *stationary* over the stream, and are unable to detect when the

distribution "drifts" or abruptly "shifts" away from the base, resulting in incorrect low-dimensional mappings (see Fig. 1).

The focus of this paper is two-fold. We first provide theoretical results that show that the quality[1] of a manifold, as learnt by Isomap, asymptotically converges as the data size, n, increases. This is a necessary result to show the correctness of streaming methods such as S-Isomap and S-Isomap++, under the assumption of stationarity. Next, we propose a methodology to detect changes in the underlying distribution of the stream properties (drifts and shifts), and inform the streaming methods to update the base manifold.

We employ a Gaussian Process (GP) [7] based adaptation of Isomap to process high-throughput streams. The use of GP is enabled by a kernel that measures the relationship between a pair of observations along the manifold, and not in the original high-dimensional space. We prove that the low-dimensional representations inferred using the GP based method – *GP-Isomap* – are equivalent to the representations obtained using the state-of-art streaming Isomap methods [5, 6]. Additionally, we empirically show, on synthetic and real data sets, that the predictive variance associated with the GP predictions is an effective indicator of the changes (either gradual drifts or sudden shifts) in the underlying generative distribution, and can be employed to inform the algorithm to "re-learn" the core manifold.

Related works

Processing data streams efficiently using standard approaches is challenging in general, given streams require real-time processing and cannot be stored permanently. Any form of analysis, including detecting concept drift, requires adequate summarization which can deal with the inherent constraints and that can approximate the characteristics of the stream well. Sampling based strategies include *random sampling* [8, 9] as well as decision-tree based approaches [10] which have been used in this context. To identify concept drift, maintaining statistical summaries on a streaming "window" is a typical strategy [11–13]. However, none of these are applicable in the setting of learning a latent representation from the data, e.g., manifolds, in the presence of changes in the stream distribution.

We discuss limitations of existing incremental and streaming solutions that have been specifically developed in the context of manifold learning, specifically in the context of the Isomap algorithm in . Coupling Isomap with GP Regression (GPR) has been explored in the past [14–17], though not in the context of streaming data. This includes a Mercer kernel-based Isomap technique [14] and an emulator pipeline using Isomap to determine a low-dimensional representation, whose output is fed to a GPR model [15].

The intuition to use GPR for detecting concept drift is novel even though the Bayesian non-parametric approach [18], primarily intended for anomaly detection, comes close to our work in a single manifold setting. However, their choice of the Euclidean distance (in original $\mathbb{R}^D$ space) based kernel for its covariance matrix, can result in high Procrustes error, as shown in Fig. 4. Additionally, their approach does not scale, given it does not use any approximation to be able to process the new streaming points "cheaply".

---

[1] See Sect. "Problem statement and preliminaries" for the definition of manifold quality.

We also note that a family of GP based non-spectral[2] non-linear dimensionality reduction methods exist, called *Gaussian Process Latent Variable Model* (GPLVM) [20] and its variants [19, 21]. GPLVM assumes that the high-dimensional observations are generated from the corresponding low-dimensional representations, using a GP prior. The latent low-dimensional representations are then inferred by maximizing the marginalized log-likelihood of the observed data, which is an optimization problem with n unknown d-dimensional vectors, where d is the length of the low-dimensional representation. In contrast, the GP-Isomap algorithm assumes that the low-dimensional representations are generated from the corresponding high-dimensional data, using a manifold-specific kernel matrix.

There has been a considerable body of literature dealing with dimensionality reduction [22, 23], including recent work that uses deep learning based models [24], however, these cannot be applied in a streaming setting. While there have been some recent works that use PCA in a streaming setting [25], these are inherently linear and hence are not applicable where the manifolds are non-linear.

Problem statement and preliminaries

We first formulate the NLDR problem and provide background on Isomap and discuss its out-of-sample and streaming extensions [5, 6, 26, 27]. Additionally, we provide brief introduction to Gaussian Process (GP) analysis.

Non-linear dimensionality reduction

Given high-dimensional data $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1...n}$, where $\mathbf{y}_i \in \mathbb{R}^D$, the NLDR problem is concerned with finding its corresponding low-dimensional representation $\mathbf{X} = \{\mathbf{x}_i\}_{i=1...n}$, such that $\mathbf{x}_i \in \mathbb{R}^d$, where $d \ll D$.

NLDR methods assume that the data lies along a low-dimensional manifold embedded in a high-dimensional space, and exploit the *global* (Isomap [1], Minimum Volume Embedding [28]) or *local* (LLE [2], Laplacian Eigenmaps [29], Hessian Eigenmaps [30]) properties of the manifold to map each $\mathbf{y}_i$ to its corresponding $\mathbf{x}_i$.

The Isomap algorithm [1] maps each $\mathbf{y}_i$ to its low-dimensional representation $\mathbf{x}_i$ in such a way that the geodesic distance along the manifold between any two points, $\mathbf{y}_i$ and $\mathbf{y}_j$, is as close to the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ as possible. The geodesic distance is approximated by computing the shortest path between the two points using the k-nearest neighbor graph[3] and is stored in the geodesic distance matrix $\mathbf{G} = \{\mathbf{g}_{i,j}\}_{1 \leq i,j \leq n}$, where $\mathbf{g}_{i,j}$ is the geodesic distance between the points $\mathbf{y}_i$ and $\mathbf{y}_j$. $\widetilde{\mathbf{G}} = \{\mathbf{g}_{i,j}^2\}_{1 \leq i,j \leq n}$ contains squared geodesic distance values. The Isomap algorithm recovers $\mathbf{x}_i$ by using the classical *Multi Dimensional Scaling* (MDS) on $\widetilde{\mathbf{G}}$. Let $\mathbf{B}$ be the inner product matrix between different $\mathbf{x}_i$. $\mathbf{B}$ can be retrieved as $\mathbf{B} = -\mathbf{H}\widetilde{\mathbf{G}}\mathbf{H}/2$ by assuming $\sum_{i=1}^{n} \mathbf{x}_i = 0$, where $\mathbf{H} = \{\mathbf{h}_{i,j}\}_{1 \leq i,j \leq n}$ and $\mathbf{h}_{i,j} = \delta_{i,j} - 1/n$, where $\delta_{i,j}$ is the Kronecker delta. Isomap uncovers $\mathbf{X}$ such that $\mathbf{X}^T\mathbf{X}$ is as close to $\mathbf{B}$ as possible. This is

---

[2] An equivalence between GPLVM and Kernel Principal Component Analysis (KPCA) has been shown in the literature [19].

[3] Actually, there are two variants of Isomap. The former employs a **K**-rule to define the neighborhood $\mathcal{N}(\mathbf{y})$ for each point $\mathbf{y} \in \mathbf{Y}$ i.e. it considers the k-nearest neighbors of each point $\mathbf{y}$ to be its neighborhood $\mathcal{N}(\mathbf{y})$. The second variant employs a $\boldsymbol{\epsilon}$-rule to define the neighborhood $\mathcal{N}(\mathbf{y})$ of $\mathbf{y}$ i.e. it considers all points which are within a radius of $\boldsymbol{\epsilon}$ to be in its neighborhood $\mathcal{N}(\mathbf{y})$. We observe that there is a direct one-to-one relationship between the two rules with regards to computing the neighborhood $\mathcal{N}(\mathbf{y})$ for all $\mathbf{y} \in \mathbf{Y}$.

achieved by setting $\mathbf{X} = \{\sqrt{\lambda_1}\mathbf{q}_1 \ \sqrt{\lambda_2}\mathbf{q}_2 \ \ldots \ \sqrt{\lambda_d}\mathbf{q}_d\}^T$ where $\lambda_1, \lambda_2 \ldots \lambda_d$ are the d largest eigenvalues of $\mathbf{B}$ and $\mathbf{q}_1, \mathbf{q}_2 \ldots \mathbf{q}_d$ are the corresponding eigenvectors.

The Isomap algorithm makes use of $\widetilde{\mathbf{G}}$ to approximate the pairwise Euclidean distances on the generated manifold. Isomap demonstrates good performance when the computed geodesic distances are close to Euclidean. In this scenario, the matrix $\mathbf{B}$ behaves like a positive semi-definite (PSD) kernel. The opposite scenario requires a modification to be made to $\widetilde{\mathbf{G}}$ to make it PSD. In MDS literature, this is commonly referred to as the Additive Constant Problem (ACP) [14, 31, 32].

To measure error between the true, underlying low-dimensional representation to that uncovered by NLDR methods, *Procrustes analysis* [33] is typically used. Procrustes analysis involves aligning two matrices, $\mathbf{A}$ and $\mathbf{B}$, by finding the optimal translation, $\mathbf{t}$, rotation, $\mathbf{R}$, and scaling factor, $\mathbf{s}$, that minimizes the *Frobenius norm* between the two aligned matrices, i.e.,:

$$\epsilon_{\mathrm{Proc}}(\mathbf{A}, \mathbf{B}) = \min_{\mathbf{R},\mathbf{t},\mathbf{s}} \|\mathbf{sRB} + \mathbf{t} - \mathbf{A}\|_{\mathrm{F}} \tag{1}$$

The above optimization problem has a closed-form solution obtained by performing *Singular Value Decomposition* (SVD) of $\mathbf{AB}^T$ [33]. Consequently, one of the properties of Procrustes analysis is that $\epsilon_{\mathrm{Proc}}(\mathbf{A}, \mathbf{B}) = 0$ when $\mathbf{A} = \mathbf{sRB} + \mathbf{t}$ i.e. when one of the matrices is a scaled, translated and/or rotated version of the other, which we leverage upon in this work.

Streaming Isomap

Given that the Isomap algorithm has a complexity of $\mathcal{O}(n^3)$ (where $n = $ size of data) since it needs to perform Eigen Decomposition on $\mathbf{B}$ as described in the previous section, recomputing the manifold is computationally impractical to use in a streaming setting. Incremental techniques have been proposed in the past [5, 27], which can efficiently process the new streaming points, without affecting the quality of the embedding significantly.

The S-Isomap algorithm relies on the assumption that a stable manifold can be learnt using only a fraction of the stream (denoted as the batch data set $\mathcal{B}$), and the remaining part of stream (denoted as the stream data set $\mathcal{S}$) can be mapped to the manifold in a significantly less costly manner. A convergence proof that justifies this assumption is provided in Sect. "Convergence proofs for S-Isomap and S-Isomap++". Alternatively, this can be justified by considering the convergence of eigenvectors and eigenvalues of $\mathbf{B}$, as the number of points in the batch increase [34]. In particular, the bounds on the convergence error for a similar NLDR method, i.e., kernel PCA, is shown to be inversely proportional to the batch size [34]. Similar arguments can be made for Isomap, by considering the equivalence between Isomap and Kernel PCA [26, 35]. This relationship has also been empirically shown for multiple data sets [5]. The S-Isomap algorithm computes the low-dimensional representation for each new point i.e. $\mathbf{x}_{n+1} \in \mathbb{R}^d$ by solving a least-squares problem formulated by matching the dot product of the new point with the low-dimensional embedding of the points in the batch data set $\mathbf{X}$, computed using

Isomap, to the normalized squared geodesic distances vector $\mathbf{f}$. The least-squares problem has the following form:

$$\mathbf{X}^T\mathbf{x}_{n+1} = \mathbf{f} \tag{2}$$

where[4]

$$\mathbf{f}_i \simeq \frac{1}{2}\Big(\frac{1}{n}\sum_j \mathbf{g}_{i,j}^2 - \mathbf{g}_{i,n+1}^2\Big) \tag{4}$$

where $\mathbf{g}_{i,j}$ refer to the geodesic distance discussed in Sect. "Problem statement and preliminaries".

Handling multiple manifolds

In the ideal case, when manifolds are densely sampled and sufficiently separated, clustering can be performed before applying NLDR techniques [37, 38], by choosing an appropriate local neighborhood size so as not to include points from other manifolds and still be able to capture the local geometry of the manifold. However, if the manifolds are close or intersecting, such methods typically fail. While methods such as *Generalized Principal Component Analysis* (GPCA) [39] have been proposed to generalize linear methods such as PCA for a case where the data lies on multiple sub-spaces, such ideas have not been explored for non-linear methods.

The S-Isomap++ [6] algorithm overcomes limitations of the S-Isomap algorithm and extends it to be able to deal with multiple manifolds. It uses the notion of Multi-scale SVD [40] to define tangent manifold planes at each data point, computed at the appropriate scale, and computes similarity in a local neighborhood. Additionally, it includes a novel manifold tangent clustering algorithm to be able to deal with the above issue of clustering manifolds which are close and in certain scenarios, intersecting, using these tangent manifold planes. After initially clustering the high-dimensional batch data set, the algorithm applies NLDR on each manifold individually and eventually "stitches" them together in a global ambient space by defining transformations which can map points from the individual low-dimensional manifolds to the global space. S-Isomap++ does not assume that the number of manifolds (p) is specified and automatically infers p using its clustering mechanism.[5] Given that the data points lie on low-dimensional and potentially intersecting manifolds, it is evident that the standard clustering methods, such as K-Means [42], that operate on the observed data in $\mathbb{R}^{\mathbf{D}}$, will fail in correctly identifying the clusters.

---

[4] Note that the Incremental Isomap algorithm [27] has a slightly different formulation where

$$\mathbf{f}_i \simeq \frac{1}{2}\Big(\frac{1}{n}\sum_j \mathbf{g}_{i,j}^2 - \frac{1}{n^2}\sum_{l,m} \mathbf{g}_{l,m}^2\Big) + \frac{1}{2}\Big(\frac{1}{n}\sum_j \mathbf{g}_{j,n+1}^2 - \mathbf{g}_{i,n+1}^2\Big) \tag{3}$$

where $\mathbf{g}_{i,j}$ refer to the geodesic distance discussed in Sect. "Problem statement and preliminaries". The S-Isomap algorithm assumes that the data stream draws from an uniformly sampled, unimodal distribution $p(\mathbf{x})$ and that the stream $\mathcal{S}$ and the batch $\mathcal{B}$ data sets get generated from $p(\mathbf{x})$. Additionally it assumes that the manifold has stabilized i.e. $|\mathcal{B}| = n$ is large enough. Using these assumptions in (3) above, we have that $\Big(\frac{1}{n}\sum_j \mathbf{g}_{j,n+1}^2 - \frac{1}{n^2}\sum_{l,m} \mathbf{g}_{l,m}^2\Big) = \boldsymbol{\epsilon} \simeq 0$ i.e. the expectation of squared geodesic distances for points in the batch data set $\mathcal{B}$ is close to those for points in the stream data set $\mathcal{S}$. The line of reasoning for this follows from [36]. Thus (3) simplifies to (4).

[5] In cases of uneven/low density sampling, the clustering strategy discussed might possibly generate many small clusters. In such cases, one can try to merge clusters [41], based on their affinity/closeness to make the clusters' size reasonable.

However, S-Isomap++ can only detect manifolds which it encounters in its batch learning phase and not those which it might encounter in the streaming phase. Thus, S-Isomap++ ceases to "learn" and evolve to be able to limit the embedding error for points in the data stream, even though it has a "stitching" mechanism to embed individual low-dimensional manifolds, which might themselves be of different dimensions.

Gaussian process regression

Let us assume that we are learning a probabilistic regression model to obtain the prediction at a given test input, $\mathbf{y}$, using a non-linear and latent function, $f(\cdot)$. Assuming[6] $d = 1$, the observed output, x, is related to the input as:

$$\mathbf{x} = f(\mathbf{y}) + \boldsymbol{\varepsilon}, \text{ where, } \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \boldsymbol{\sigma}_n^2) \tag{5}$$

Given a training set of inputs, $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1\dots n}$ and corresponding outputs, $\mathbf{X} = \{\mathbf{x}_i\}_{i=1\dots n}$,[7] the Gaussian Process Regression (GPR) model assumes a GP prior on the latent function values, i.e., $f(\mathbf{y}) \sim GP(m(\mathbf{y}), k(\mathbf{y}, \mathbf{y}'))$, where $m(\mathbf{y})$ is the mean of $f(\mathbf{y})$ and $k(\mathbf{y}, \mathbf{y}')$ is the covariance between any two evaluations of $f(\cdot)$, i.e, $m(\mathbf{y}) = \mathbb{E}[f(\mathbf{y})]$ and $k(\mathbf{y}, \mathbf{y}') = \mathbb{E}[(f(\mathbf{y}) - m(\mathbf{y}))(f(\mathbf{y}') - m(\mathbf{y}'))]$. Here we use a zero-mean function ($m(\mathbf{y}) = 0$), though other functions could be used as well. The GP prior states that any finite collection of the latent function evaluations are jointly Gaussian, i.e.,

$$f(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n) \sim \mathcal{N}(\mathbf{0}, K) \tag{6}$$

where the ij^{th} entry of the $n \times n$ covariance matrix, K, is given by $k(\mathbf{y}_i, \mathbf{y}_j)$. The GPR model uses (5) and (6) to obtain the predictive distribution at a new test input, $\mathbf{y}_{n+1}$, as a Gaussian distribution with following mean and variance:

$$\mathbb{E}[\mathbf{x}_{n+1}] = \mathbf{k}_{n+1}^\top (K + \sigma_n^2 I)^{-1} \mathbf{X} \tag{7}$$

$$var[\mathbf{x}_{n+1}] = k(\mathbf{y}_{n+1}, \mathbf{y}_{n+1}) - \mathbf{k}_{n+1}^\top (K + \sigma_n^2 I)^{-1} \mathbf{k}_{n+1} + \sigma_n^2 \tag{8}$$

where $\mathbf{k}_{n+1}$ is a $n \times 1$ vector with i^{th} value as $k(\mathbf{y}_{n+1}, \mathbf{y}_i)$.

The kernel function, $k(\cdot)$, specifies the covariance between function values, $f(\mathbf{y}_i)$ and $f(\mathbf{y}_j)$, as a function of the corresponding inputs, $\mathbf{y}_i$ and $\mathbf{y}_j$. A popular choice is the *squared exponential* kernel, which has been used in this work:

$$k(\mathbf{y}_i, \mathbf{y}_j) = \sigma_s^2 \exp\left[-\frac{\|\mathbf{y}_i - \mathbf{y}_j\|^2}{2\ell^2} \right] \tag{9}$$

where σ_s^2 is the signal variance and ℓ is the length scale. The quantities σ_s^2, ℓ, and σ_n^2 from (5) are the hyper-parameters of the model and can be estimated by maximizing the marginal log-likelihood of the observed data ($\mathbf{Y}$ and $\mathbf{X}$) under the GP prior assumption.

---

[6] For vector-valued outputs, i.e., $\mathbf{x} \in \mathbb{R}^d$, one can consider d independent models.

[7] While the typical notation for GPR models uses $\mathbf{X}$ as inputs and $\mathbf{Y}$ as outputs [7], we have reversed the notation to maintain consistency with rest of the paper.

One can observe that predictive mean, $\mathbb{E}[\mathbf{x}_{n+1}]$ in (7) can be written as an inner product, i.e.

$$\mathbb{E}[\mathbf{x}_{n+1}] = \boldsymbol{\beta}^\top \mathbf{k}_{n+1} \tag{10}$$

where $\boldsymbol{\beta} = (K + \sigma_n^2 I)^{-1}\mathbf{X}$. We will utilize this form in subsequent proofs.

Convergence proofs for S-isomap and S-isomap++

In this section, we demonstrate the convergence of the S-Isomap algorithm for a single manifold setting, subsequent to which we extend it to the multi-manifold setting i.e. for the S-Isomap++ algorithm described above.

Theorem 1 *Given a uniformly sampled, uni-modal distribution from which the random batch data set $\mathcal{B} = \{\mathbf{y}_i \in \mathbb{R}^{\mathbf{D}}\}_{i=1\dots n}$ of the S-Isomap algorithm is derived from, there exists a threshold $\mathbf{n}_0$, such that when $\mathbf{n} \geq \mathbf{n}_0$, the Procrustes Error $\epsilon_{\mathrm{Proc}}(\boldsymbol{\tau}_{\mathcal{B}}, \boldsymbol{\tau}_{\mathrm{ISO}})$ between $\boldsymbol{\tau}_{\mathcal{B}} = \boldsymbol{\phi}^{-1}(\mathcal{B})$, the true underlying representation and $\boldsymbol{\tau}_{\mathrm{ISO}} = \hat{\boldsymbol{\phi}}^{-1}(\mathcal{B})$, the embedding uncovered by Isomap is small ($\epsilon_{\mathrm{Proc}} \approx 0$) i.e. the batch phase of the S-Isomap algorithm converges, where $\boldsymbol{\phi}(\cdot)$ is the non-linear function which maps data points from the underlying low-dimensional ground truth representation $\mathbf{U}$ to $\mathcal{B} \in \mathbb{R}^{\mathbf{D}}$ and the ground truth $\mathbf{U}$ originally resides in a convex $\mathbb{R}^{\mathbf{d}}$ Euclidean space.*

Proof Based on the setting described above, the S-Isomap algorithm acts like a *generative model* which is trying to learn the inverse mapping $\boldsymbol{\phi}(\cdot)^{-1}$, where the associated embedding error is the Procrustes Error $\epsilon_{\mathrm{Proc}}(\boldsymbol{\tau}_{\mathcal{B}}, \boldsymbol{\tau}_{\mathrm{ISO}})$.

The proof follows from [43] who showed that in a setting, where given $\lambda_1, \lambda_2, \mu > 0$ and for appropriately chosen $\epsilon > 0$, as well as a data set $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1\dots n}$ sampled from a Poisson distribution with density function $\boldsymbol{\alpha}$ which satisfies the $\boldsymbol{\delta}$-sampling condition i.e.

$$\boldsymbol{\alpha} > \log(\mathbf{V}/(\boldsymbol{\mu}\widetilde{\mathbf{V}}(\delta/4)))/\widetilde{\mathbf{V}}(\delta/2) \tag{11}$$

wherein the ϵ-rule is used to construct a graph $\mathbf{G}$ on $\mathbf{Y}$, the ratio between the graph based distance $\mathbf{d}_G(\mathbf{x}, \mathbf{y})$ and the true Euclidean distance $\mathbf{d}_M(\mathbf{x}, \mathbf{y})$ $\forall \mathbf{x}, \mathbf{y} \in \mathbf{Y}$ is bounded. More concretely, the following holds with probability at least $(1 - \boldsymbol{\mu})$ for $\forall \mathbf{x}, \mathbf{y} \in \mathbf{Y}$:

$$1 - \lambda_1 \leq \frac{\mathbf{d}_G(\mathbf{x}, \mathbf{y})}{\mathbf{d}_M(\mathbf{x}, \mathbf{y})} \leq 1 + \lambda_2 \tag{12}$$

where $\mathbf{V}$ is the volume of the manifold $\mathcal{M}$ and

$$\widetilde{\mathbf{V}}(\mathbf{r}) = \min_{\mathbf{x} \in \mathcal{M}} \mathrm{Vol}\,(\mathcal{B}_{\mathbf{x}}(\mathbf{r})) = \boldsymbol{\eta}_{\mathbf{d}} \mathbf{r}^{\mathbf{d}} \tag{13}$$

is the volume of the smallest metric ball in $\mathcal{M}$ of radius $\mathbf{r}$ and $\boldsymbol{\delta} > 0$ is such that

$$\boldsymbol{\delta} = \lambda_2 \boldsymbol{\epsilon}/4 \tag{14}$$

A similar result can be derived in the scenario where $\mathbf{n}$ points are sampled independently from the fixed probability distribution $p(\mathbf{y}; \boldsymbol{\theta})$, in which case we have :

$$\mathbf{n}\widetilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha} \tag{15}$$

where $\widetilde{\boldsymbol{\alpha}}$ is the probability of selecting a sample from $p(\mathbf{y}; \boldsymbol{\theta})$.
Using (13), (14) and (15) in (11), we have:

$$\begin{aligned} \mathbf{n}\widetilde{\boldsymbol{\alpha}} &> \log(\mathbf{V}/(\mu \widetilde{\mathbf{V}}(\delta/4)))/\widetilde{\mathbf{V}}(\delta/2) \\ &= \left[\log(\mathbf{V}/\mu \eta_d(\lambda_2 \epsilon/16)^{\mathbf{d}}) \right]/\eta_d(\lambda_2 \epsilon/8)^{\mathbf{d}} \end{aligned} \tag{16}$$

$$\begin{aligned} \mathbf{n} &> (1/\widetilde{\boldsymbol{\alpha}}) \left[\log(\mathbf{V}/\mu \eta_d(\lambda_2 \epsilon/16)^{\mathbf{d}}) \right]/\eta_d(\lambda_2 \epsilon/8)^{\mathbf{d}} \\ &= \mathbf{n}_0 \end{aligned} \tag{17}$$

where $\mathbf{n}_0 = (1/\widetilde{\boldsymbol{\alpha}}) \left[\log(\mathbf{V}/\mu \eta_d(\lambda_2 \epsilon/16)^{\mathbf{d}}) \right]/\eta_d(\lambda_2 \epsilon/8)^{\mathbf{d}}$, is the condition which ensures that (12) is satisfied.

Thus we have an adequate threshold for the size of the batch data set $\mathcal{B}$ which ensures (17) is satisfied for the ϵ-rule. We can derive a similar threshold for the $\mathbf{K}$-rule, observing that there is a direct one-to-one mapping between $\mathbf{K}$ and ϵ (See Sect. "Non-linear dimensionality reduction" for more details).

To complete the proof, we observe that (12) implies that $\mathbf{d}_G(\mathbf{x}, \mathbf{y})$, the graph based distance between points $\mathbf{x}, \mathbf{y} \in \mathbf{G}$ is a perturbed version of $\mathbf{d}_M(\mathbf{x}, \mathbf{y})$, the true Euclidean distance between points $\mathbf{x}$ and $\mathbf{y}$ in $\mathbb{R}^{\mathbf{d}}$. Let $\widetilde{\mathbf{D}}_M$ and $\widetilde{\mathbf{D}}_G$ represent the squared distance matrix corresponding to $\mathbf{d}_M(\mathbf{x}, \mathbf{y})$ and $\mathbf{d}_G(\mathbf{x}, \mathbf{y})$ respectively. Thus we have $\widetilde{\mathbf{D}}_G = \widetilde{\mathbf{D}}_M + \Delta\widetilde{\mathbf{D}}_M$ where $\Delta\widetilde{\mathbf{D}}_M = \{\Delta\widetilde{\mathbf{d}}_M(\mathbf{i}, \mathbf{j})\}_{1 \le i,j \le n}$ and $\Delta\widetilde{\mathbf{d}}_M(\mathbf{i}, \mathbf{j})$ are bounded due to (12).

In the past [44], the robustness of MDS to small perturbations was demonstrated as follows. Let $\mathbf{F}$ represent the zero-diagonal symmetric matrix which perturbs the true squared distance matrix $\mathbf{B}$ to $\mathbf{B} + \Delta\mathbf{B} = \mathbf{B} + \epsilon\mathbf{F}$. Then the Procrustes Error between the embeddings uncovered by MDS for $\mathbf{B}$ and for $\mathbf{B} + \Delta\mathbf{B}$ is given by $\frac{\epsilon^2}{4} \sum_{j,k} \frac{\mathbf{e}_j^T \mathbf{F} \mathbf{e}_k^2}{\lambda_j + \lambda_k}$, which is very small for small entries $\{\mathbf{f}_{i,j}\}_{1 \le i,j \le n} \in \mathbf{F}$, $\{\mathbf{e}_k(\lambda_k)\}_{k=1 \dots n}$ represent the eigenvectors (eigenvalues) of $\mathbf{B}$ and the double summation is over pairs of $(\mathbf{j}, \mathbf{k}) = 1, 2, \dots (\mathbf{n} - 1)$ but excluding those pairs $(\mathbf{j}, \mathbf{k})$ wherein both entries of which lie in the range $(\mathbf{K} + 1), (\mathbf{K} + 2), \dots (\mathbf{n} - 1)$, $\mathbf{K} = \sum_{k=1}^{n} \mathcal{I}(\lambda_k > 0)$ and $\mathcal{I}(\cdot)$ is the indicator function. We substitute $\epsilon = 1$ and replace $\mathbf{B}$ with $\widetilde{\mathbf{D}}_M$ and $\Delta\mathbf{B}$ with $\Delta\widetilde{\mathbf{D}}_M$ above to complete the proof, since the entries of $\Delta\widetilde{\mathbf{D}}_M$ are very small i.e. $\{0 \le \Delta\mathbf{d}_M(i,j) \le \lambda^2\}_{1 \le i,j \le n}$ where $\lambda = \max(\lambda_1, \lambda_2)$ for small λ_1, λ_2, given the condition $\mathbf{n} > \mathbf{n}_0$ is satisfied for (12). Thus we have that the embedding uncovered by S-Isomap for a batch data set $\mathcal{B}$ where $|\mathcal{B}| = \mathbf{n} > \mathbf{n}_0$ converges asymptotically to their true embedding upto translation, rotation and scaling factors. $\qquad\square$

Extension to the multi-manifold setting

The above proof can be extended to show the convergence of the S-Isomap++ [6] algorithm, described in Sect. "Handling Multiple Manifolds" as follows.

Corollary 1 *The batch phase of the S-Isomap++ algorithm converges under appropriate conditions.*

Proof Similar to the proof for the S-Isomap algorithm, we consider a corresponding setting for the multi-manifold scenario now, wherein we are attempting to learn the inverse mappings $\boldsymbol{\phi}(\cdot)_{i=1,\dots,\mathbf{p}}^{-1}$ for each of the $\mathbf{p}$ manifolds. The initial clustering step of the S-Isomap++ algorithm separates the samples from the batch data set $\mathcal{B}$ into different individual clusters $\mathcal{B}_i$, such that each cluster is mutually exclusive of the others and corresponds to one of the multiple manifolds present in the data i.e. $\bigcup\limits_{i=1}^{\mathbf{p}} \mathcal{B}_i = \mathcal{B}$ and $\mathcal{B}_i \bigcap\limits_{\forall i,j, i \neq j} \mathcal{B}_j = \phi$.

The intuition for clustering and subsequently processing each of the clusters separately is based on the setting described above that the observed data was generated by first sampling points from multiple $\mathbf{U}_{i=1,\dots,\mathbf{p}}$ i.e., convex domains in $\mathbb{R}^{\mathbf{d}}$[8] and subsequently mapping those points in non-linear fashion, using possibly different $\boldsymbol{\phi}(\cdot)_{i=1,\dots,\mathbf{p}}$ to $\mathcal{B} \in \mathbb{R}^{\mathbf{D}}$. Thus, to learn the different inverse mappings effectively, there is a need to be able to cluster the data appropriately.

After the initial clustering step, a similar analysis as in Theorem 1 provides thresholds $\mathbf{n}_i, \forall i \in \{1, \dots, \mathbf{p}\}$ for each of the $\mathbf{p}$ clusters beyond which when $|\mathcal{B}_i| = \mathbf{n} \geq \mathbf{n}_i$, the Procrustes Error $\epsilon_{\text{Proc}}(\boldsymbol{\tau}_{\mathcal{B}_i}, \boldsymbol{\tau}_{\text{ISO}_i})$ between $\boldsymbol{\tau}_{\mathcal{B}_i} = \boldsymbol{\phi}_i^{-1}(\mathcal{B}_i)$, the true underlying representation and $\boldsymbol{\tau}_{\text{ISO}_i} = \hat{\boldsymbol{\phi}}_i^{-1}(\mathcal{B}_i)$, the embedding uncovered by Isomap is small ($\epsilon_{\text{Proc}} \approx 0$) i.e. the batch phase of the S-Isomap++ algorithm converges provided each of the $\mathbf{p}$ clusters $\mathcal{B}_{i=1,\dots,\mathbf{p}}$ exceeds the appropriate threshold $\mathbf{n}_i$ (similar to (17) above). □

The S-Isomap++ algorithm does not assume that the number of manifolds ($\mathbf{p}$) is specified. Refer to Sect. "Handling multiple manifolds" for more details.

Methodology

The proposed GP-Isomap algorithm follows a two-phase strategy (similar to the S-Isomap and S-Isomap++), where exact manifolds are learnt from an initial batch $\mathcal{B}$, and subsequently a computationally inexpensive mapping procedure processes the remainder of the stream. To handle multiple manifolds, the batch data $\mathcal{B}$ is first clustered via manifold tangent clustering or other standard techniques. Exact Isomap is applied on each cluster. The resulting low-dimensional data for the clusters is then "stitched" together to obtain the low-dimensional representation of the input data. The difference from the past methods is the mapping procedure which uses GPR to obtain the predictions for the low-dimensional mapping (see (7)). At the same time, the associated predictive variance (see (8)) is used to detect changes in the underlying distribution.

---

[8] It is possible that the low-dimensional Euclidean space specific to each manifold is different i.e. $\mathbf{U}_i$ is a convex domain in $\mathbb{R}^{\mathbf{d}_i}$ space, where $\mathbf{d}_i \neq \mathbf{d}_j$. However we can imagine a scenario where we choose a $\mathbb{R}^{\mathbf{d}}$ global space, where $\mathbf{d} = \sum_i \mathbf{d}_i$ from which the different convex $\mathbf{U}_i$ were sampled from. Additionally note that convexity is preserved by linear projections to higher dimensional spaces thus the convex domains $\mathbf{U}_{i=1,\dots,p}$ remain convex in this new space.

The overall GP-Isomap algorithm is outlined in 1 and takes a batch data set, $\mathcal{B}$ and the streaming data, $\mathcal{S}$ as inputs, along with other parameters. The processing is split into two phases: a batch learning phase (Lines 1–15) and a streaming phase (Lines 16–32), which are described later in this section.

Algorithm 1 GP-Isomap

Require: Batch data set: $\mathcal{B}$, Streaming data set: $\mathcal{S}$; Parameters: ϵ, k, l, λ, σ_t, n_s
Ensure: $\mathcal{Y}_\mathcal{S}$: low-dimensional representation for $\mathcal{S}$
    ▷ Batch Phase
1: $\mathcal{C}_{i=1,2...p} \leftarrow \text{FIND\_CLUSTERS}(\mathcal{B}, \epsilon)$
2: $\boldsymbol{\xi}_s \leftarrow \emptyset$
3: **for** $1 \leq i \leq p$ **do**
4:    $\mathcal{LDE}_i, \mathcal{G}_i \leftarrow \text{ISOMAP}(\mathcal{C}_i)$
5: **end for**
6: **for** $1 \leq i \leq p$ **do**
7:    $\phi_i^{\mathcal{GP}} \leftarrow \text{ESTIMATE}(\mathcal{LDE}_i, \mathcal{G}_i)$
8: **end for**
9: $\boldsymbol{\xi}_s \leftarrow \bigcup\limits_{i=1}^{p} \bigcup\limits_{j=i+1}^{p} \text{NN}(\mathcal{C}_i, \mathcal{C}_j, \boldsymbol{k}) \cup \text{FN}(\mathcal{C}_i, \mathcal{C}_j, \boldsymbol{l})$
10: $\mathcal{GE}_s \leftarrow \text{MDS}(\boldsymbol{\xi}_s)$
11: **for** $1 \leq j \leq p$ **do**
12:    $\mathcal{I} \leftarrow \boldsymbol{\xi}_s \cap \mathcal{C}_j$
13:    $\mathcal{A} \leftarrow \begin{bmatrix} \mathcal{LDE}_j^{\mathcal{I}} \\ e^T \end{bmatrix}$
14:    $\mathcal{R}_i, t_i \leftarrow \mathcal{GE}_{\mathcal{I},s} \times \mathcal{A}^T (\mathcal{A}\mathcal{A}^T + \boldsymbol{\lambda} I)^{-1}$
15: **end for**
    ▷ Streaming Phase
16: $\mathcal{S}_u \leftarrow \emptyset$
17: **for** $s \in \mathcal{S}$ **do**
18:    **if** $|\mathcal{S}_u| \geq n_s$ **then**
19:      $\mathcal{Y}_u \leftarrow$ Re-run Batch Phase with $\mathcal{B} \leftarrow \mathcal{B} \cup \mathcal{S}_u$
20:    **end if**
21:    **for** $1 \leq i \leq p$ **do**
22:      $\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i \leftarrow \text{GPR}(s, \mathcal{LDE}_i, \mathcal{G}_i, \phi_i^{GP})$
23:    **end for**
24:    $j \leftarrow \text{argmin}_i |\boldsymbol{\sigma}_i|$
25:    **if** $\boldsymbol{\sigma}_j \leq \sigma_t$ **then**
26:      $y_s \leftarrow \mathcal{R}_j \boldsymbol{\mu}_j + t_j$
27:      $\mathcal{Y}_\mathcal{S} \leftarrow \mathcal{Y}_\mathcal{S} \cup y_s$
28:    **else**
29:      $\mathcal{S}_u \leftarrow \mathcal{S}_u \cup \boldsymbol{s}$
30:    **end if**
31: **end for**
32: **return** $\mathcal{Y}_\mathcal{S}$

Kernel function

The key innovation here is to use a manifold-specific kernel matrix in the GPR method. The matrix **B**, which is the inner product matrix between the points in the low-dimensional space (see Sect. "Non-linear dimensionality reduction"), could be a reasonable starting point. However, as past researchers have shown [16], typical kernels, such as squared exponential kernel, can only be generalized to a positive definite kernel on a geodesic metric space if the space is flat. Thus **B** will not necessarily yield a valid positive semi-definite kernel matrix. However, a result by [32] shows that a small positive constant, $\lambda_{\max}$, can be added to **B** to guarantee that it will be PSD. This constant can be calculated as the largest eigenvalue of the matrix:

$$\mathbf{M} = \begin{bmatrix} 0 & 2\mathbf{B} \\ -\mathbf{I} & -4\mathbf{P} \end{bmatrix} \tag{18}$$

where $\mathbf{P} = -\mathbf{HGH}/2$. Here, **G** is the geodesic distance matrix and $\mathbf{H} = \{\mathbf{h}_{i,j}\}_{1 \leq i,j \leq n}$, $\mathbf{h}_{i,j} = \delta_{i,j} - 1/n$, where $\delta_{i,j}$ is the Kronecker delta. $\widetilde{\mathbf{B}}$ can be derived from **B** as [32]:

$$\widetilde{\mathbf{B}} = \mathbf{B} + 2\lambda_{\max}\mathbf{P} + \frac{1}{2}\lambda_{\max}^2\mathbf{H} \tag{19}$$

where $\lambda_{\max}$ is the largest eigenvalue of **M**.

The proposed GP-Isomap algorithm uses a novel geodesic distance based kernel function defined as:

$$k(\mathbf{y}_i, \mathbf{y}_j) = \sigma_s^2 \exp\left(-\frac{\widetilde{\mathbf{b}}_{i,j}}{2\ell^2}\right) \tag{20}$$

where $\widetilde{\mathbf{b}}_{i,j}$ is the ij^{th} entry of the matrix $\widetilde{\mathbf{B}}$, σ_s^2 is the signal variance (whose value we fix as 1 in this work) and ℓ is the length scale hyper-parameter. Thus the kernel matrix $\mathbf{K}$ can be written as:

$$\mathbf{K} = \exp\left(-\frac{\widetilde{\mathbf{B}}}{2\ell^2}\right) \tag{21}$$

This kernel function plays a key role in using the GPR model for mapping streaming points on the learnt manifold, by measuring similarity along the low-dimensional manifold, instead of the original space ($\mathbb{R}^D$), as is typically done in GPR based solutions.

The matrix $\widetilde{\mathbf{B}}$, is positive semi-definite. Consequently, we note that the kernel matrix, $\mathbf{K}$, is positive definite (refer (22) below).

Using 1, the novel kernel we propose can be written as

$$\mathbf{K}(\mathbf{x}, \mathbf{y}) = \mathbf{I} + \sum_{i=1}^{d}\left[\exp\left(-\frac{\lambda_i}{2\ell^2}\right) - 1\right]\mathbf{q}_i\mathbf{q}_i^T = \mathbf{I} + \mathbf{Q}\widetilde{\mathbf{\Lambda}}\mathbf{Q}^T \tag{22}$$

where $\widetilde{\mathbf{\Lambda}} = \begin{bmatrix} \left[\exp\left(-\frac{\lambda_1}{2\ell^2}\right) - 1\right] & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \left[\exp\left(-\frac{\lambda_d}{2\ell^2}\right) - 1\right] \end{bmatrix}$ and $\{\lambda_i, \mathbf{q}_i\}_{i=1\dots d}$ are eigenvalue/

eigenvector pairs of $\widetilde{\mathbf{B}}$ as discussed in Sect. "Non-linear dimensionality reduction".

Batch learning

The batch learning phase consists of these tasks:

i). **Clustering**: The first step in the batch phase involves clustering of the batch data set $\mathcal{B}$ into $\mathbf{p}$ individual clusters which represent the manifolds (Line 1). In case, $\mathcal{B}$ contains a single cluster, the algorithm can correctly detect it. Refer to Sect. "Handling multiple manifolds" for more details,

ii). **Dimensionality reduction**: Subsequently, full Isomap is executed on each of the $\mathbf{p}$ individual clusters to get low-dimensional representations $\mathcal{LDE}_{i=1,2\dots\mathbf{p}}$ of the data points belonging to each individual cluster (Lines 3–5),

iii). **Hyper-parameter estimation**: The geodesic distance matrix for the points in the ith manifold $\mathcal{G}_i$ and the corresponding low-dimensional representation $\mathcal{LDE}_i$, are fed to the GP model for each of the $\mathbf{p}$ manifolds, to perform hyper-parameter estimation, which outputs $\{\boldsymbol{\phi}_i^{GP}\}_{i=1,2\dots\mathbf{p}}$ (Lines 6–8), and,

iv). **Learning mapping to global space**: The low-dimensional embedding uncovered for each of the manifolds can be of different dimensionalities. Consequently, a mapping to a unified global space is needed. To learn this mapping, a support set $\boldsymbol{\xi}_s$ is formulated, which contains the $\mathbf{k}$ pairs of nearest points and $\mathbf{l}$ pairs of farthest points, between each pair of manifolds. Subsequently, MDS is executed on

this support set $\boldsymbol{\xi}_s$ to uncover its low-dimensional representation $\mathcal{GE}_s$. Individual scaling and translation factors $\{\mathcal{R}_i, t_i\}_{i=1,2\ldots\mathbf{p}}$ are learnt via solving a least squares problem involving $\boldsymbol{\xi}_s$, which map points from each of the individual manifolds to the global space (Lines 9–15).

Stream processing

In the streaming phase, each sample $\mathbf{s}$ in the stream set $\mathcal{S}$ is embedded using each of the $\mathbf{p}$ GP models to evaluate the prediction $\boldsymbol{\mu}_i$, along with the variance $\boldsymbol{\sigma}_i$ (Lines 22–24). The manifold with the smallest variance get chosen to embed the sample $\mathbf{s}$ into, using the corresponding scaling $\mathcal{R}_j$ and translation factor t_j, provided $min_i|\boldsymbol{\sigma}_i|$ is within the allowed threshold $\boldsymbol{\sigma}_t$ (Lines 25–28), otherwise sample s is added to the unassigned set $\mathcal{S}_u$ (Lines 29–31). When the size of unassigned set $\mathcal{S}_u$ exceeds certain threshold $\mathbf{n}_s$, we add them to the batch data set and re-learn the base manifold (Line 18–20). The assimilation of the new points in the batch maybe done more efficiently in an incremental manner.

Complexity

The runtime complexity of our proposed algorithm is dominated by the GP regression step as well as the Isomap execution step, both of which have $\mathcal{O}(n^3)$ complexity, where n is the size of the batch data set $\mathcal{B}$. This is similar to the S-Isomap and S-Isomap++ algorithms, that also have a runtime complexity of $\mathcal{O}(n^3)$. The stream processing step is $\mathcal{O}(n)$ for each incoming streaming point. The space complexity of GP-Isomap is dominated by $\mathcal{O}(n^2)$. This is because each of the samples of the stream set $\mathcal{S}$ get processed separately. Thus, the space requirement as well as runtime complexity does not grow with the size of the stream, which makes the algorithm appealing for handling high-volume streams.

Theoretical analysis

We first state the main result for the single manifold case, and prove it using results in the Appendix section, and then present proofs for the multi-manifold case.

Theorem 2   *For a single manifold setting, the prediction $\boldsymbol{\tau}_{\mathrm{GP}}$ of GP-Isomap is equivalent to the prediction $\boldsymbol{\tau}_{\mathrm{ISO}}$ of S-Isomap, i.e., the Procrustes Error $\boldsymbol{\epsilon}_{\mathrm{Proc}}\big(\boldsymbol{\tau}_{\mathrm{GP}}, \boldsymbol{\tau}_{\mathrm{ISO}}\big)$ between $\boldsymbol{\tau}_{\mathrm{GP}}$ and $\boldsymbol{\tau}_{\mathrm{ISO}}$ is* 0.

Proof The prediction of GP-Isomap is given by (10). Using Lemma 5, we note that

$$\boldsymbol{\beta} = \left\{ \frac{\boldsymbol{\alpha}\sqrt{\lambda}_1 \mathbf{q}_1}{1 + \boldsymbol{\alpha}\mathbf{c}_1} \frac{\boldsymbol{\alpha}\sqrt{\lambda}_2 \mathbf{q}_2}{1 + \boldsymbol{\alpha}\mathbf{c}_2} \cdots \frac{\boldsymbol{\alpha}\sqrt{\lambda}_d \mathbf{q}_d}{1 + \boldsymbol{\alpha}\mathbf{c}_d} \right\} \tag{23}$$

The term $\mathbf{K}_*$ for GP-Isomap, using the novel kernel function evaluates to:

$$\mathbf{K}_* = \exp\left(-\frac{\mathbf{G}_*^2}{2\ell^2}\right) \tag{24}$$

where $\mathbf{G}_*^2$ represents the vector containing the squared geodesic distances of $\mathbf{x}_{n+1}$ to $\mathbf{X}$ containing $\{\mathbf{x}_i\}_{i=1,2\ldots n}$.

Considering the above equation element-wise, we note that the $\mathbf{i}^{\text{th}}$ term of $\mathbf{K}_*$ equates to $\exp\left[-\frac{\mathbf{g}_{i,n+1}^2}{2\ell^2}\right]$. Using Taylor's series expansion we have,

$$\exp\left[-\frac{\mathbf{g}_{i,n+1}^2}{2\ell^2}\right] \simeq \left(1 - \frac{\mathbf{g}_{i,n+1}^2}{2\ell^2}\right) \text{ for large } \ell \tag{25}$$

The prediction by the S-Isomap is given by (4) as follows:

$$\boldsymbol{\tau}_{\text{ISO}} = \{\sqrt{\lambda_1}\mathbf{q}_1^T\mathbf{f} \ \ \sqrt{\lambda_2}\mathbf{q}_2^T\mathbf{f} \ \ldots \ \sqrt{\lambda_d}\mathbf{q}_d^T\mathbf{f}\}^T \tag{26}$$

where $\mathbf{f} = \{\mathbf{f}_i\}$ is as defined by (4).
Rewriting (4) we have:

$$\mathbf{f}_i \simeq \frac{1}{2}\left(\boldsymbol{\gamma} - \mathbf{g}_{i,n+1}^2\right) \tag{27}$$

where $\boldsymbol{\gamma} = \left(\frac{1}{n}\sum_j \mathbf{g}_{i,j}^2\right)$ is a constant with respect to $\mathbf{x}_{n+1}$, since it depends only on squared geodesic distance values associated within the batch data set $\mathcal{B}$ and $\mathbf{x}_{n+1}$ is part of the stream data set $\mathcal{S}$.

We now consider the 1st dimension of the predictions for GP-Isomap and S-Isomap only and demonstrate their equivalence via Procrustes Error. The analysis for the remaining dimensions follows a similar line of reasoning.

Thus for the 1st dimension, using (27) the S-Isomap prediction is:

$$\begin{aligned}
\boldsymbol{\tau}_{\text{ISO}_1} &= \sqrt{\lambda_1}\mathbf{q}_1^T\mathbf{f} \\
&= \sqrt{\lambda_1}\sum_{i=1}^{n}\mathbf{q}_{1,i}\left(\frac{1}{2}\left(\boldsymbol{\gamma} - \mathbf{g}_{i,n+1}^2\right)\right) \\
&= \frac{\sqrt{\lambda_1}}{2}\sum_{i=1}^{n}\mathbf{q}_{1,i}\left(\boldsymbol{\gamma} - \mathbf{g}_{i,n+1}^2\right)
\end{aligned} \tag{28}$$

Similarly using Lemma 5, (24) and (25), we have that the $\mathbf{1}^{\text{st}}$ dimension for GP-Isomap prediction is given by,

$$\begin{aligned}
\boldsymbol{\tau}_{\text{GP}_1} &= \frac{\alpha\sqrt{\lambda_1}\mathbf{q}_1^T}{1+\alpha c_1}\mathbf{K}_* \\
&= \frac{\alpha\sqrt{\lambda_1}}{1+\alpha c_1}\sum_{i=1}^{n}\mathbf{q}_{1,i}\left(1 - \frac{\mathbf{g}_{i,n+1}^2}{2\ell^2}\right)
\end{aligned} \tag{29}$$

We can observe that $\boldsymbol{\tau}_{\text{GP}_1}$ is a scaled and translated version of $\boldsymbol{\tau}_{\text{ISO}_1}$. Similarly for each of the dimensions ($1 \leq i \leq d$), the prediction for the GP-Isomap $\boldsymbol{\tau}_{\text{GP}_i}$ can be shown to be a scaled and translated version of the prediction for the S-Isomap $\boldsymbol{\tau}_{\text{ISO}_i}$. These individual

scaling $\mathbf{s}_i$ and translation $\mathbf{t}_i$ factors can be represented together by single collective scaling $\mathbf{s}$ and translation $\mathbf{t}$ factors. Consequently, the Procrustes Error $\epsilon_{\text{Proc}}(\boldsymbol{\tau}_{\text{GP}}, \boldsymbol{\tau}_{\text{SI}})$ is 0. (refer Sect. "Non-linear dimensionality reduction"). $\square$

Results and analysis

In this section, we demonstrate the performance of the proposed algorithm on both synthetic and real-world data sets. In Sect. "Results on synthetic data sets", we present results for synthetic data sets, whereas Sect. "Results on sensor data set" contains results on benchmark sensor data sets. All experiments were done using Python 3.0 implementations of the proposed and related methods, and were run on a MacBook Pro (2.8 GHz Quad-Core Intel Core i7, 16 GB 1600 MHz DDR3). Our results demonstrate that: (i). GP-Isomap is able to perform good quality dimension reduction on a manifold, (ii). the reduction produced by GP-Isomap is equivalent to the corresponding output of S-Isomap (or S-Isomap++), and (iii). the predictive variance within GP-Isomap is able to identify changes in the underlying distribution in the data stream on all data sets considered in this paper.

In the interest of space, we avoid comparing the quality of the dimensionality reduction with other methods, and refer readers to S-Isomap and S-Isomap++ where these equivalent methods were shown to be better than existing approaches for dimensionality reduction.

GP-Isomap has the following hyper-parameters: ϵ, k, l, λ, σ_t, n_s. We set k, l, λ to have values of 16, 1 and 0.005, respectively. We study the effect of σ_t and n_s using the different data sets listed in Sects. "Results on synthetic data sets" and "Results on sensor data set" respectively.

Results on synthetic data sets

Swiss roll data sets are typically used for evaluating manifold learning algorithms. To evaluate our method on concept drift, we use the Euler Isometric Swiss Roll data set [5] consisting of four $\mathbb{R}^2$ Gaussian patches having $n = 2000$ points each, chosen at random, which are embedded into $\mathbb{R}^3$ using a non-linear function $\boldsymbol{\psi}(\cdot)$. The points for each of the Gaussian modes were divided equally into training and test sets randomly. To test incremental concept drift, we use one of the training data sets from the above data set, along with a uniform distribution of points for testing (refer to Fig. 1 for details). Figures 2a and 3a demonstrate our results on this data set.

To evaluate our method on sudden concept drift, we trained our GP-Isomap model using the first three out of four training sets of the Euler Isometric Swiss Roll data set. Subsequently we stream points randomly from the test sets from only the first three classes initially and later stream points from the test set of the fourth class, keeping track of the predictive variance all the while. Figure 2a demonstrates the sudden increase (see red line) in the variance of the stream when streaming points are from the fourth class i.e. unknown mode. Thus GP-Isomap is able to detect concept drift correctly, and is able to map all of the data points correctly on the lower dimensional manifold, as shown in Fig. 3a. The bottom panel of Fig. 1 demonstrates the performance of S-Isomap++ on this data set. It fails to map the streaming points of the unknown mode correctly, given it had not encountered the unknown mode during the batch training phase.

(a) Euler Isometric Swiss Roll

(b) Gas Sensor Array Drift

(c) Human Activity Recognition
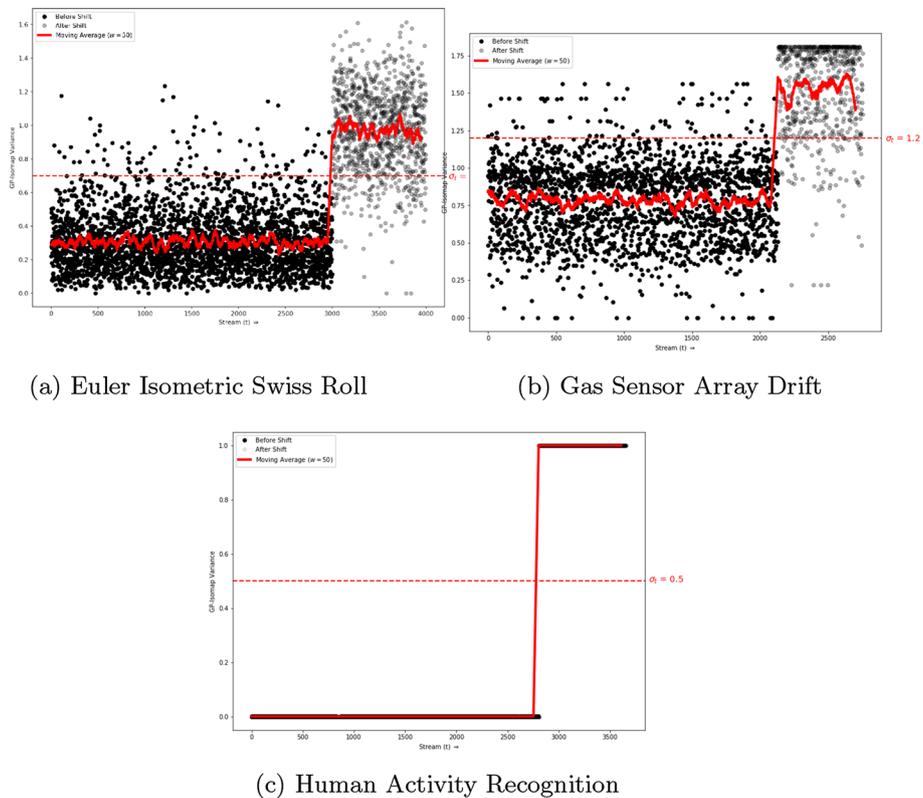
Fig. 2 Using variance to detect concept drift for different data sets. The *x*-axis represents time and the *y*-axis represents the model's predictive variance for the stream. Initially, when stream consists of samples generated from known modes, variance is low. Later, when samples from an unrecognized mode appear, variance drastically shoots up. For the first two data sets, noisy instances in the initial part get assigned a large variance, sporadically. The variance is well-behaved for the third data set. The optimal values of hyper-parameters, n_s and σ_t, were set to (1000, 0.7), (412, 1.2), (855, 0.5), for the three data sets

In Sect. "Theoretical analysis", we proved the equivalence between the prediction of S-Isomap with that of GP-Isomap, using our novel kernel. In Fig. 4, we show empirically via Procrustes Error (PE) that the prediction of S-Isomap matches that of GP-Isomap, irrespective of size of batch used. PE for GP-Isomap with the Euclidean distance based kernel remains high irrespective of the size of the batch, which clearly demonstrates the unsuitability of this kernel to adequately learn mappings in the low-dimensional space.

Results on sensor data set

In this section, we present results from different benchmark sensor data sets to demonstrate the efficacy of our algorithm.

Results on gas sensor array drift data set

The *Gas Sensor Array Drift* [45] data set is a benchmark data set ($n = 13910$) available to research communities to develop strategies to dealing with concept drift and uses measurements from 16 chemical sensors used to discriminate between 6 gases

(a) Euler Isometric Swiss Roll data set

(b) Gas Sensor Array Drift data set

(c) Human Activity Recognition data set
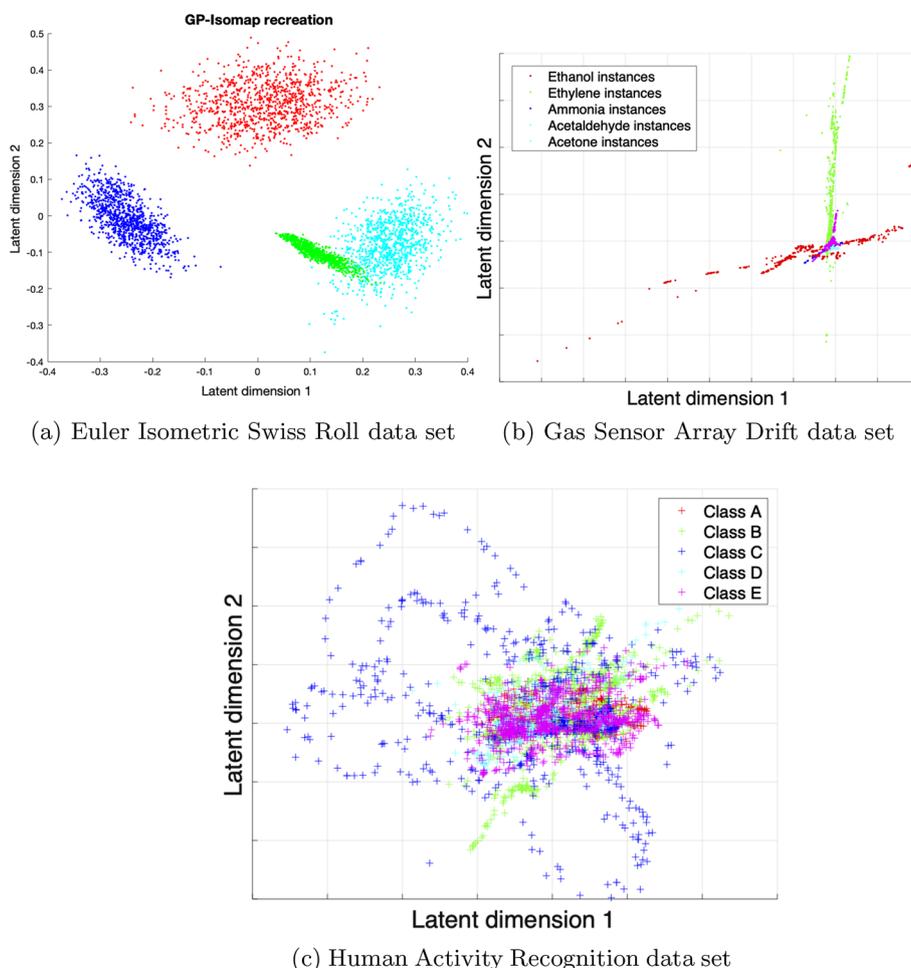
Fig. 3 Low dimensional representations uncovered by GP-Isomap for three different data sets. For the Swiss roll data, GP-isomap is able to learn the structure in the data using a 2-D reduction, while for the real-world census data sets, the structure is not evident in 2-D and possibly a higher dimensional manifold is required

(class labels) at various concentrations. We demonstrate the performance of our proposed method on this data set.

We first remove instances which had invalid/empty entries as feature values. Subsequently the data is *mean normalized*. Data points from the first five classes were divided into training and test sets. We train our model using the training data from four out of these five classes. While testing, we stream points randomly from the test sets of these four classes first and later stream points from the test set of the fifth class. Figures 2b and 3b demonstrate our results on this data set. From Fig. 2b, we observe that our model can clearly detect concept drift due to the unknown fifth class by tracking the variance of the stream, using the running average (red line). However, as shown in Fig. 3b, a two-dimensional manifold is not sufficient to capture the cluster structure in the data set.
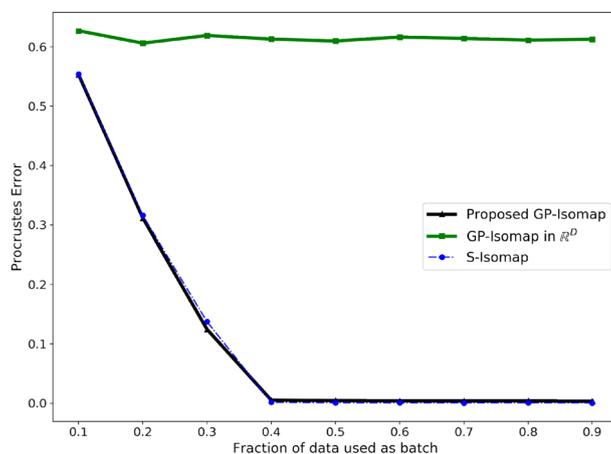
Fig. 4 Procrustes error (PE) between the ground truth with **a** GP-Isomap (blue line) with the geodesic distance based kernel, **b** S-Isomap (dashed blue line with dots) and **c** GP-Isomap (green line) using the Euclidean distance based kernel, for different fractions (*f*) of data used in the batch $\mathcal{B}$. The behavior of PE for **a** closely matches that for **b**. However, the PE for GP-Isomap using the Euclidean distance kernel remains high irrespective of *f* demonstrating its unsuitability for manifolds

Results on human activity recognition (HAR) data set

The *Human Activity Recognition* [46] data set consists of multiple data sets which are focused on discriminating between different activities, i.e. to predict which activity was performed at a specific point in time. In this work, we focused on the *Weight Lifting Exercises* (WLE) data set ($n = 39242$) which investigates how well an activity was performed by the wearer of different sensor devices. The WLE data set consists of six young health participants who performed one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes.

The data set was cleaned i.e. instances with invalid/empty entries were removed. Subsequently the data points from the different classes were *mean normalized* and divided into training and test sets. Figures 2c and 3c demonstrate our results on this data set. Figure 2c demonstrates the concept drift phenomenon. Similar to the methodology we used earlier to detect concept drift, we initially trained our algorithm using instances from the latter four classes only, whereas during the streaming phase we randomly selected instances from the streaming set of these four classes first and later streamed points from the first class, keeping track of the predictive variance.

Conclusions

We have proposed a streaming Isomap algorithm (GP-Isomap) that can be used to learn non-linear low-dimensional representation of high-dimensional data arriving in a streaming fashion. This algorithm can have significant applications in areas involving analysis of high-dimensional data streams [47], especially in constrained environments, such as real-time situational monitoring [48, 49], healthcare monitoring [50], scientific

data visualization [51], etc. We prove that using a GPR formulation to map incoming data instances onto an existing manifold is equivalent to using existing geometric strategies [5, 6]. Moreover, by utilizing a small batch for the initial learning of the manifold, as well as for training the GPR model, the method scales linearly with the size of the stream, thereby ensuring its applicability for practical problems. Using the Bayesian inference of the GPR model allows us to estimate the variance associated with the mapping of the streaming instances. The variance is shown to be a strong indicator of changes in the underlying stream properties on a variety of data sets. By utilizing the variance, one can devise re-training strategies that can include expanding the batch data set. While in the experiments we have demonstrated the ability of GP-Isomap to detect shifts in the underlying distributions, the algorithm can also be used to detect gradual shifts, as illustrated in Fig. 1. While we have focused on Isomap algorithm in this paper, similar formulations can be applied for other NLDR methods such as LLE [2], Laplacian Eigenmaps [29], Hessian Eigenmaps [30], etc., and will be explored as future research.

Appendix

Lemma 1   *The matrix exponential for* $\mathbf{M}$ *for* $\text{rank}(\mathbf{M}) = d$ *and symmetric* $\mathbf{M}$ *is given by*

$$e^{\mathbf{M}} = \mathbf{I} + \sum_{i=1}^{d} \left(e^{\lambda_i} - 1\right)\mathbf{q}_i\mathbf{q}_i^{\top}$$

where $\{\lambda_i\}_{i=1,2\ldots d}$ *are the d largest eigenvalues of* $\mathbf{M}$ *and* $\{\mathbf{q}_i\}_{i=1,2\ldots d}$ *are the corresponding eigenvectors such that* $\mathbf{q}_i^{\top}\mathbf{q}_j = \delta_{i,j}$.

Proof   Let $\mathbf{M}$ be an $n \times n$ real matrix. The exponential $e^{\mathbf{M}}$ is given by

$$e^{\mathbf{M}} = \sum_{k=0}^{\infty} \frac{1}{k!}\mathbf{M}^{\mathbf{k}} = \mathbf{I} + \sum_{k=1}^{\infty} \frac{1}{k!}\mathbf{M}^{\mathbf{k}}$$

where $\mathbf{I}$ is the identity. Real, symmetric $\mathbf{M}$ has real eigenvalues and mutually orthogonal eigenvectors i.e. $\mathbf{M} = \sum_{i=1}^{n} \lambda_i\mathbf{q}_i\mathbf{q}_i^{\top}$ where $\{\lambda_i\}_{i=1\ldots n}$ are real and $\mathbf{q}_i^{\top}\mathbf{q}_j = \delta_{i,j}$. Given $\mathbf{M}$ has rank d, we have $\mathbf{M} = \sum_{i=1}^{d} \lambda_i\mathbf{q}_i\mathbf{q}_i^{\top}$.

$$
\begin{aligned}
e^{\mathbf{M}} &= \mathbf{I} + \sum_{i=1}^{\infty} \frac{1}{i!} \mathbf{M}^i \\
&= \mathbf{I} + \frac{1}{1!} \left(\lambda_1 \mathbf{q}_1 \mathbf{q}_1^\top + \lambda_2 \mathbf{q}_2 \mathbf{q}_2^\top + \ldots + \lambda_d \mathbf{q}_d \mathbf{q}_d^\top \right) \\
&\quad + \frac{1}{2!} \left(\lambda_1 \mathbf{q}_1 \mathbf{q}_1^\top + \lambda_2 \mathbf{q}_2 \mathbf{q}_2^\top + \ldots + \lambda_d \mathbf{q}_d \mathbf{q}_d^\top \right)^2 + \ldots \\
&= \mathbf{I} + \left(\frac{\lambda_1}{1!} + \frac{\lambda_1^2}{2!} + \ldots \right) \mathbf{q}_1 \mathbf{q}_1^\top + \left(\frac{\lambda_2}{1!} + \frac{\lambda_2^2}{2!} + \ldots \right) \mathbf{q}_2 \mathbf{q}_2^\top + \ldots \\
&\quad + \left(\frac{\lambda_d}{1!} + \frac{\lambda_d^2}{2!} + \ldots \right) \mathbf{q}_d \mathbf{q}_d^\top \\
&= \mathbf{I} + (e^{\lambda_1} - 1) \mathbf{q}_1 \mathbf{q}_1^\top + (e^{\lambda_2} - 1) \mathbf{q}_2 \mathbf{q}_2^\top + \ldots + (e^{\lambda_d} - 1) \mathbf{q}_d \mathbf{q}_d^\top \\
&= \mathbf{I} + \sum_{i=1}^{d} (e^{\lambda_i} - 1) \mathbf{q}_i \mathbf{q}_i^\top
\end{aligned}
\tag{30}
$$

$\square$

Lemma 2 *The inverse of the Gaussian kernel for* $rank(\mathbf{M}) = 1$ *and symmetric* $\mathbf{M}$ *is given by*

$$
\left(\mathbf{K} + \sigma_n^2 \mathbf{I} \right)^{-1} = \alpha \mathbf{I} - \frac{\alpha^2 \mathbf{c}_1 \mathbf{q}_1 \mathbf{q}_1^\top}{1 + \alpha \mathbf{c}_1}
$$

where $\mathbf{q}_1$ *is the first eigenvector of* $\mathbf{M}$ *i.e.* $\mathbf{q}_1^\top \mathbf{q}_1 = 1$, λ_1 *is the corresponding eigenvalue and* $\alpha = \frac{1}{(1 + \sigma_n^2)}$ *and* $\mathbf{c}_1 = \left[\exp \left(-\frac{\lambda_1}{2\ell^2} \right) - 1 \right]$.

Proof Using (22) for $d = 1$, we have

$$
\begin{aligned}
\left(\mathbf{K} + \sigma_n^2 \mathbf{I} \right)^{-1} &= \left(\mathbf{I} + \left[\exp \left(-\frac{\lambda_1}{2\ell^2} \right) - 1 \right] \mathbf{q}_1 \mathbf{q}_1^\top + \sigma_n^2 \mathbf{I} \right)^{-1} \\
&= \left((1 + \sigma_n^2) \mathbf{I} + \left[\exp \left(-\frac{\lambda_1}{2\ell^2} \right) - 1 \right] \mathbf{q}_1 \mathbf{q}_1^\top \right)^{-1}
\end{aligned}
\tag{31}
$$

Representing $\frac{1}{(1 + \sigma_n^2)}$ as α and $\left[\exp \left(-\frac{\lambda_1}{2\ell^2} \right) - 1 \right]$ as $\mathbf{c}_1$ and using $(1 + \sigma_n^2) \mathbf{I}$ as $\mathbf{A}$, $\mathbf{c}_1 \mathbf{q}_1$ as $\mathbf{u}$ and $\mathbf{q}_1$ as $\mathbf{v}$ in the Sherman-Morrison identity [52], we have

$$
\begin{aligned}
\left(\mathbf{K} + \sigma_n^2 \mathbf{I} \right)^{-1} &= \alpha \mathbf{I} - \frac{\alpha \mathbf{I} \mathbf{c}_1 \mathbf{q}_1 \mathbf{q}_1^\top \alpha \mathbf{I}}{1 + \alpha \mathbf{c}_1} \\
&= \alpha \mathbf{I} - \frac{\alpha^2 \mathbf{c}_1 \mathbf{q}_1 \mathbf{q}_1^\top}{1 + \alpha \mathbf{c}_1}
\end{aligned}
\tag{32}
$$

$\square$

Lemma 3 *The inverse of the Gaussian kernel for* $rank(\mathbf{M}) = d$ *and symmetric* $\mathbf{M}$ *is given by*

$$\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1} = \alpha\mathbf{I} - \alpha^2 \sum_{i=1}^{d} \frac{c_i\mathbf{q}_i\mathbf{q}_i^\top}{1 + \alpha c_i}$$

where $\{\lambda_i\}_{i=1,2\dots d}$ *are the d largest eigenvalues of* $\mathbf{M}$ *and* $\{\mathbf{q}_i\}_{i=1,2\dots d}$ *are the corresponding eigenvectors such that* $\mathbf{q}_i^\top\mathbf{q}_j = \delta_{i,j}$.

Proof Using the result of previous lemma iteratively, we get the required result

$$\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1} = \alpha\mathbf{I} - \alpha^2 \sum_{i=1}^{d} \frac{c_i\mathbf{q}_i\mathbf{q}_i^\top}{1 + \alpha c_i} \tag{33}$$

where $\alpha = \frac{1}{\left(1+\sigma_n{}^2\right)}$ and $c_i = \left[\exp\left(-\frac{\lambda_i}{2\ell^2}\right) - 1\right]$.   □

Lemma 4 *The solution for Gaussian Process regression system, for the scenario when* $rank(\mathbf{M}) = 1$ *and for symmetric* $\mathbf{M}$ *is given by*

$$\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1}\mathbf{y} = \frac{\alpha\sqrt{\lambda_1}\mathbf{q}_1}{1 + \alpha c_1}$$

Proof Assuming the intrinsic dimensionality of the low-dimensional manifold to be 1 implies that the inverse of the Gaussian kernel is as defined as in (32). $\mathbf{y}$ is $\sqrt{\lambda_1}\mathbf{q}_1$ in this case (refer Sect. "Non-linear dimensionality reduction"). Thus we have

$$\begin{aligned}
\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1}\mathbf{y} &= \left(\alpha\mathbf{I} - \frac{\alpha^2 c_1\mathbf{q}_1\mathbf{q}_1^\top}{1 + \alpha c_1}\right)\left(\sqrt{\lambda_1}\mathbf{q}_1\right) \\
&= \alpha\sqrt{\lambda_1}\mathbf{q}_1 - \frac{\alpha^2\sqrt{\lambda_1}c_1\mathbf{q}_1}{1 + \alpha c_1} = \frac{\alpha\sqrt{\lambda_1}\mathbf{q}_1}{1 + \alpha c_1}
\end{aligned} \tag{34}$$

□

Lemma 5 *The solution for Gaussian Process regression system, for the scenario when* $rank(\mathbf{M}) = d$ *and for symmetric* $\mathbf{M}$ *is given by*

$$\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1}\mathbf{y} = \left\{\frac{\alpha\sqrt{\lambda_1}\mathbf{q}_1}{1 + \alpha c_1} \frac{\alpha\sqrt{\lambda_2}\mathbf{q}_2}{1 + \alpha c_2} \dots \frac{\alpha\sqrt{\lambda_d}\mathbf{q}_d}{1 + \alpha c_d}\right\}$$

Proof Assuming the intrinsic dimensionality of the low-dimensional manifold to be d implies that the inverse of the Gaussian kernel is as defined as in (33). $\mathbf{y}$ is $\{\sqrt{\lambda_1}\mathbf{q}_1 \ \sqrt{\lambda_2}\mathbf{q}_2 \ \dots \ \sqrt{\lambda_d}\mathbf{q}_d\}$ in this case (refer Sect. "Non-linear dimensionality reduction"), where $\mathbf{q}_i^\top\mathbf{q}_j = \delta_{i,j}$. Each of the k dimensions of $\left(\mathbf{K} + \sigma_n{}^2\mathbf{I}\right)^{-1}\mathbf{y}$ can be processed independently, similar to the previous lemma. For the ith dimension, we have,

$$\left(\mathbf{K} + {\sigma_n}^2\mathbf{I}\right)^{-1}\mathbf{y}_i = \left(\alpha\mathbf{I} - \alpha^2\sum_{j=1}^{d}\frac{\mathbf{c}_j\mathbf{q}_j\mathbf{q}_j^{\top}}{1 + \alpha\mathbf{c}_j}\right)\left(\sqrt{\lambda_i}\mathbf{q}_i\right)$$

$$= \alpha\sqrt{\lambda_i}\mathbf{q}_i - \alpha^2\sum_{j=1}^{d}\frac{\mathbf{c}_j\mathbf{q}_j\mathbf{q}_j^{\top}\mathbf{q}_i\left(\sqrt{\lambda_i}\right)}{1 + \alpha\mathbf{c}_j} \tag{35}$$

$$= \alpha\sqrt{\lambda_i}\mathbf{q}_i - \frac{\alpha^2\sqrt{\lambda_i}\mathbf{c}_i\mathbf{q}_i}{1 + \alpha\mathbf{c}_i} = \frac{\alpha\sqrt{\lambda_i}\mathbf{q}_i}{1 + \alpha\mathbf{c}_i}$$

Thus we get the result,

$$\left(\mathbf{K} + \sigma_n^2\mathbf{I}\right)^{-1}\mathbf{y} = \{\frac{\alpha\sqrt{\lambda_1}\mathbf{q}_1}{1 + \alpha\mathbf{c}_1} \ \frac{\alpha\sqrt{\lambda_2}\mathbf{q}_2}{1 + \alpha\mathbf{c}_2} \ \cdots \ \frac{\alpha\sqrt{\lambda_d}\mathbf{q}_d}{1 + \alpha\mathbf{c}_d}\} \tag{36}$$

$\square$

Declarations

Ethics approval and consent to participate
The author confirms the sole responsibility for this manuscript. The author read and approved the final manuscript.

Consent for publication
The authors consent for publication.

Competing interests
The authors declare that they have no competing interests.

References
1. Tenenbaum JB, De Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. Science. 2000;290(5500):2319–23.
2. Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. Science. 2000;290(5500):2323–6.
3. Silva VD, Tenenbaum JB. Global versus local methods in nonlinear dimensionality reduction. NeurIPS. 2003;721–728.
4. Wu Y, Chan KL. An extended isomap algorithm for learning multi-class manifold. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics. 2004:6;3429–3433.
5. Schoeneman F, Mahapatra S, Chandola V, Napp N, Zola J. Error metrics for learning reliable manifolds from streaming data. In: SDM. 2017:750–758. SIAM
6. Mahapatra S, Chandola V. S-isomap++: Multi manifold learning from streaming data. In: 2017 IEEE International Conference on Big Data (Big Data). 2017:716–725.
7. Williams CK, Seeger M. Using the nyström method to speed up kernel machines. In: NeurIPS. 2001:682–688.
8. Vitter JS. Random sampling with a reservoir. ACM Trans Math Softw (TOMS). 1985;11(1):37–57.
9. Chaudhuri S, Motwani R, Narasayya V. On random sampling over joins. ACM SIGMOD Record. 1999;28:263–74.
10. Domingos P, Hulten G. Mining high-speed data streams. Kdd. 2000;2:4.
11. Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. J Comput Syst Sci. 1999;58(1):137–47.

12. Jagadish HV, Koudas N, Muthukrishnan S, Poosala V, Sevcik KC, Suel T. Optimal histograms with quality guarantees. VLDB. 1998;98:24–7.
13. Datar M, Gionis A, Indyk P, Motwani R. Maintaining stream statistics over sliding windows. SIAM J Comput. 2002;31(6):1794–813.
14. Choi H, Choi S. Kernel isomap. Electron Lett. 2004;40(25):1612–3.
15. Xing W, Shah AA, Nair PB. Reduced dimensional gaussian process emulators of parametrized partial differential equations based on isomap. Proc Royal Soc A Math Phys Eng Sci. 2015;471(2174):20140697.
16. Feragen A, Lauze F, Hauberg S. Geodesic exponential kernels: when curvature and linearity conflict. New Orleans: IEEE CVPR; 2015. p. 3032–42.
17. Chapelle O, Haffner P, Vapnik VN. Support vector machines for histogram-based image classification. IEEE Trans Neural Netw. 1999;10(5):1055–64.
18. Barkan O, Weill J, Averbuch A. Gaussian process regression for out-of-sample extension. 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing. 2016.
19. Li P, Chen S. A review on gaussian process latent variable models. CAAI Trans Intell Technol. 2016;1(4):366–76.
20. Lawrence ND. Gaussian process latent variable models for Visualisation of high dimensional data. NeurIPS. 2003;16:329–36.
21. Titsias M, Lawrence ND. Bayesian gaussian process latent variable model. AISTATS. 2010;9:844–51.
22. Henriksen A, Ward R. Adaoja: adaptive learning rates for streaming PCA. CoRR arXiv.1905.12115. https://doi.org/10.48550/arXiv.1905.12115.
23. Rani R, Khurana M, Kumar A, Kumar N. Big data dimensionality reduction techniques in iot: review, applications and open research challenges. Cluster Computing 2022.
24. Kiarashinejad Y, Abdollahramezani S, Adibi A. Deep learning approach based on dimensionality reduction for designing electromagnetic nanostructures. npj Comput Mater. 2020;6(1):12.
25. Balzano L, Chi Y, Lu YM. Streaming pca and subspace tracking: the missing data case. Proc IEEE. 2018;106(8):1293–310. https://doi.org/10.1109/JPROC.2018.2847041.
26. Bengio Y, Paiement J-f, Vincent P, Delalleau O, Roux NL, Ouimet M. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. NeurIPS. 2004:177–184.
27. Law MH, Jain AK. Incremental nonlinear dimensionality reduction by manifold learning. IEEE Trans Pattern Anal Mach Intell. 2006;28(3):377–91.
28. Weinberger KQ, Packer B, Saul LK. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. AISTATS. 2005;2:6.
29. Belkin M, Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering. NeurIPS, 2002:585–591.
30. Donoho DL, Grimes C. Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. Proc Natl Acad Sci. 2003;100(10):5591–6.
31. Torgerson WS. Multidimensional scaling: I. theory and method. Psychometrika. 1952;17(4):401–19.
32. Cailliez F. The analytical solution of the additive constant problem. Psychometrika. 1983;48(2):305–8.
33. Dryden IL. Shape analysis. Wiley Stats Ref: Statistics reference online; 2014.
34. Shawe-Taylor J, Williams CK. The stability of kernel principal components analysis and its relation to the process eigenspectrum. 2003:383–390.
35. Ham JH, Lee DD, Mika S, Schölkopf B. A kernel view of the dimensionality reduction of manifolds. Dep Pap (ESE). 2004;93.
36. Hoeffding W. Probability inequalities for sums of bounded random variables. J Am Stat Assoc. 1963;58(301):13–30.
37. Polito M, Perona P. Grouping and dimensionality reduction by locally linear embedding. NeurIPS. 2002:1255–1262.
38. Fan M, Qiao H, Zhang B, Zhang X. Isometric multi-manifold learning for feature extraction. In: 2012 IEEE 12th International Conference on Data Mining, 2012:241–250. IEEE
39. Vidal R, Ma Y, Sastry S. Generalized principal component analysis (gpca). IEEE Trans Pattern Anal Mach Intell. 2005;27(12):1945–59.
40. Little AV, Lee J, Jung Y-M, Maggioni M. Estimation of intrinsic dimensionality of samples from noisy low-dimensional manifolds in high dimensions with multiscale svd. In: 2009 IEEE/SP 15th Workshop on Statistical Signal Processing, 2009:85–88. IEEE.
41. Comaniciu D, Meer P. Mean shift: a robust approach toward feature space analysis. IEEE Trans Pattern Anal Mach Intell. 2002;5:603–19.
42. Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM Comput Surveys (CSUR). 1999;31(3):264–323.
43. Bernstein M, De Silva V, Langford JC, Tenenbaum JB. Graph approximations to geodesics on embedded manifolds. Citeseer: Technical report. 2000.
44. Sibson R. Studies in the robustness of multidimensional scaling: perturbational analysis of classical scaling. J Royal Stat Soc Series B Methodol. 1979;41(2):217–29.
45. Vergara A, Vembu S, Ayhan T, Ryan MA, Homer ML, Huerta R. Chemical gas sensor drift compensation using classifier ensembles. Sens Actuators B Chem. 2012;166:320–9.
46. Velloso E, Bulling A, Gellersen H, Ugulino W, Fuks H. Qualitative activity recognition of weight lifting exercises. In: Proceedings of the 4th Augmented Human International Conference. 2013:116–123. ACM
47. Gomes HM, Read J, Bifet A, Barddal JP, Gama JA. Machine learning for streaming data: State of the art, challenges, and opportunities. SIGKDD Explor Newsl. 2019:6–22.
48. Thudumu S, Branch P, Jin J, Singh JJ. A comprehensive survey of anomaly detection techniques for high dimensional big data. J Big Data. 2020;7(1):42.
49. Fujiwara T, Chou J, Shilpika S, Xu P, Ren L, Ma K. An incremental dimensionality reduction method for visualizing streaming multidimensional data. IEEE Trans Visualization Comput Graphics. 2020;26(01):418–28.

50. Gupta V, Mittal M. QRS complex detection using STFT, chaos analysis, and PCA in standard and real-time ECG databases. J Inst Eng India Series B. 2019;100(5):489–97. https://doi.org/10.1007/s40031-019-00398-9.
51. Dorier M, Wang Z, Ayachit U, Snyder S, Ross R, Parashar M. Colza: Enabling elastic in situ visualization for high-performance computing simulations. In: 2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS). 2022;538–548.
52. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical recipes in C++. Art Scientific Computi. 1992;2:1002.

Publisher's Note