Probabilistic Temporal Subspace Clustering

Behnam Gholami  
Department of Computer Science  
Rutgers University  
bb510@cs.rutgers.edu

Vladimir Pavlovic  
Department of Computer Science  
Rutgers University  
vladimir@cs.rutgers.edu

Abstract

Subspace clustering is a common modeling paradigm used to identify constituent modes of variation in data with locally linear structure. These structures are common to many problems in computer vision, including modeling time series of complex human motion. However classical subspace clustering algorithms learn the relationships within a set of data without considering the temporal dependency and then use a separate clustering step (e.g., spectral clustering) for final segmentation. Moreover, these frequently optimization-based, algorithms assume that all observations have complete features. In contrast in real-world applications, some features are often missing, which results in incomplete data and substantial performance degeneration of these approaches. In this paper, we propose a unified non-parametric generative framework for temporal subspace clustering to segment data drawn from a sequentially ordered union of subspaces that deals with the missing features in a principled way. The non-parametric nature of our generative model makes it possible to infer the number of subspaces and their dimension automatically from data. Experimental results on human action datasets demonstrate that the proposed model consistently outperforms other state-of-the-art subspace clustering approaches.

1. Introduction

High dimensional data are ubiquitous in many machine learning applications. Modeling such data using low dimensional representations can potentially reduce the computation time and memory requirements of the algorithms used to extract information from the data. A standard assumption in many applications is that high dimensional data lie on the union of a small number of much lower dimensional subspaces. The goal of subspace clustering is to simultaneously cluster data points into multiple subspaces and find the corresponding subspace for each cluster.

Mathematically, subspace clustering (SC) [7,27,45,60] is defined as follows: Let \( X \in \mathbb{R}^{d \times N} \) be the data matrix consisting of \( N \) data points \( \{x_n \in \mathbb{R}^d\}_{n=1}^N \) assumed be drawn from a union of \( S \) linear subspaces \( \Phi_s \) of unknown dimension \( K_s = \text{dim}(\Phi_s) \), with \( 0 < K_s < d \). The subspace clustering attempts to infer the number of subspaces \( S \), the subspaces \( \{\Phi_s\}_{s=1}^S \), their dimensions \( \{K_s\}_{s=1}^S \), and the clustering of the data points \( x_n \) into these subspaces.

Subspace clustering has achieved outstanding performance in many machine learning applications, such as face clustering [61], motion segmentation [8,23], document clustering [34], etc. and many SC algorithms have been developed, including Sparse and Low Rank Methods [7,24,29,54], algebraic methods [42], and statistical methods [2,13,21,49].

Recent work on low rank representation (LRR) [24–26, 30], sparse representation (SSC) [6,7], least square regression (LSR) [29], and their extensions [1,3–5,9,11,12,14–16,18,20,22,28,31–33,35,37,38,47,48,50,52,55–59,62] have attracted much attention in subspace clustering. Low-rank/spARSE methods attempt to find a new representation \( Z \in \mathbb{R}^{K \times N} \) of the data and then apply a spectral clustering method on the learned representation \( Z \). Sparse subspace clustering (SSC) algorithms [6,7] enforce a sparsity constraint on the representation \( Z \) to recover the multi-subspace structure. Low-rank representation (LRR) algorithms [24,26] impose low rank constraint on \( Z \) and least-square regression (LSR) [29] uses \( l_2 \) norm regularizer for \( Z \).

Statistical methods [2,13,21,49] usually model the data points using a mixture of probabilistic PCAs. Due to the probabilistic nature of the statistical methods, they are more robust to noise and outliers, in contrast to the low-rank/sparse methods.

One of the shortcomings of all of the above methods is that they generally assume all data points are drawn independently from multiple subspaces. Hence, they fail to exploit the information explicitly encoded into the time series. For example, in a video sequence, where the goal is to cluster the frames that belong to the same scene, it is reasonable to assume that the consecutive frames belong to one and the same scene, until a scene change occurs forming temporally consistent clusters.
Very recently, some subspace clustering methods have been proposed [23, 40, 52] that can take advantage of order information embedded in the data points to improve the clustering performance. Wu et al. [52] proposed a SC algorithm for sequential data by imposing a quadratic normalizer on the sparse representations that not only maintains the sparsity of the learned representation, but also forces the consecutive frames to have similar representation. Motivated by the well-known Laplacian regularization technique, Li et al. [23] proposed a temporal subspace clustering (TSC) method by introducing a $l_{1,2}$ norm as a regularizer for the sparse representations that not only maintains the sparsity of the learned representation, but also forces the consecutive frames to have similar representation. Motivated by the well-known Laplacian regularization technique, Li et al. [23] proposed a temporal subspace clustering (TSC) method that uses a temporal Laplacian regularization function to encode the sequential relationships in time-dependent data. They also learn a non-negative dictionary from the data rather than using the data itself as the dictionary to obtain more expressive coding.

There are two major problems with the above temporal subspace clustering methods. First, these methods are not designed to deal with missing features in a principled way. More precisely, when some entries of the data points are corrupted or missing (e.g., commonly some marker sets of multiple body parts are missing during motion capture in motion segmentation applications), these methods cannot explicitly and efficiently deal with the corrupted data. Second, the performance of these optimization-based (non-probabilistic) methods depends on a set of free parameters that need to be carefully tuned using cross-validation or other parameter tuning techniques, which increases both the computational complexity and the sensitivity of these methods.

To address the above-mentioned problems, we propose a unified probabilistic framework for temporal subspace clustering where temporal dependencies are modeled using Gaussian Process (GP) [51] priors (whose covariance function controls the desired dependence) on the data point’s clustering indices that can effectively deal with missing data. By employing Griffiths-Engen-McCloskey (GEM) distribution [17] defined via the stick breaking construction as the prior distribution on the clustering indices, our model is capable of inferring the number of the subspaces (clusters) automatically from the data. Moreover, by incorporating the Bernoulli process [39] into our model, we are able to concurrently learn the dimensionality of the subspaces from the data. Given a set of ordered data points, we also develop an EM algorithm to learn the complete set of parameters from the data itself.

The rest of this paper is organized as follows. We present the proposed temporal subspace clustering framework in Section 2. In Section 3, we develop an EM algorithm to learn the parameters of the proposed model. Experimental results are presented in Section 4. Finally, we conclude our work in Section 5.

2. Proposed Method

2.1. Problem Formulation

Let $X = [x_1, x_2, ..., x_N]$ be a sequence of $d$-dimensional time-series data, where the $n$-th data point $x_n (n = 1, ..., N)$ is sampled at time $t_n$. We assume that the data points are generated via a mixture of $S$ subspaces. Mathematically, each data point $x_n$ can be represented as $x_n = \Phi_s w_{s,n} + \alpha_s$, where $\Phi_s \in \mathbb{R}^{d \times K_s}$ and $\alpha_s \in \mathbb{R}^d$ specify the set of bases and the center of the $s$-subspace respectively, $w_{s,n}$ is the latent representation (projection) of $x_n$ in subspace $s$, $\alpha_s$ indicates the noise precision parameter and $c_n \in \{1, 2, ..., S\}$ is the index cluster for the data point $x_n$. By defining $\Phi = \{\Phi_s\}$ and $\mu = \{\mu_s\}$, the likelihood of $x_n$ given $\mathcal{M}$ becomes

$$p(x_n | \mathcal{M}) = \sum_{s=1}^{S} p(c_n = s)p(x_n | c_n = s, w_{s,n}, \Phi, \mu, \alpha_s),$$

where $p(c_n = s)$ encodes a mixture probability distribution over the $S$ clusters (subspaces), and $p(x_n | c_n = s, w_{s,n}, \Phi, \mu)$ is defined as

$$p(x_n | c_n = s, w_{s,n}, \Phi, \mu) = \mathcal{N}(x_n; \Phi_s w_{s,n} + \mu_s, \alpha_s^{-1}I),$$

where $I$ denotes the identity matrix of size $d$.

In the proposed model, we assume the number of subspaces $S$ and their dimensionality $\{K_s\}_{s=1}^{S}$ are unknown apriori. To address the problem of inferring $S$ and $\{K_s\}_{s=1}^{S}$, we employ the GEM distribution [17] and the Bernoulli process [39], respectively.

GEM distribution with parameter $\eta$, GEM($\eta$), can be defined as a distribution over a countably infinite number of objects (for simplicity, natural numbers $\mathbb{N} = \{1, 2, \ldots\}$) as

$$p(c = s) = \beta_s, \beta_s = v_s \prod_{l=1}^{s-1} (1-v_l), v_s \sim \text{Beta}(1, \eta), s \in \mathbb{N}$$

where $\beta_s$ is the mixing proportion defined by recursively breaking a unit-length stick into an infinite number of pieces. We use GEM($\eta$) as a prior distribution over the cluster indices, $p(c_n = s) = \beta_s$, as it apriori endows the model with a countably infinite number of subspaces. Since $\beta_s$’s decrease exponentially quickly, only a small number of subspaces will be used to fit the finite available data, with the appropriate number of subspaces automatically revealed by the data itself.

In order to infer the dimensionality $K_s$ of each subspace from the observed data, we introduce an auxiliary, latent
binary vector \( z_s \in \{0, 1\}^K \) for each subspace \( \Phi_s \), where the non-zero entries of \( z_s \) specify the bases of that subspace i.e., \( K_s = \sum_k z_{s,k} = \text{dim}(\Phi_s) \). Consequently, the model in Eq. 1 is reformulated as

\[
p(x_n|c_n = s, z_s, w_{s,n}, \Phi, \mu, \alpha_s) = N(x_n; \Phi_s(z_s \odot w_{s,n}) + \mu_s, \alpha_s^{-1} I)
\]

where \( \odot \) denotes the element-wise multiplication operator. Consequently, all data points \( \{x_n\} \) drawn from a given cluster (subspace) \( s \) share the same set of important bases of the subspace \( \Phi_s \) defined by \( z_s \), but each draw from a given cluster has unique weights \( w_{s,n} \).

Using a probabilistic hierarchical framework, we place a non-parametric prior distribution on each binary vector \( z_s \) by introducing auxiliary variables \( \{\Pi_s = \{\pi_{ks}\}_{k=1}^K\}_{s=1}^S \) drawn from the Beta distribution as

\[
\pi_{ks} \sim \text{Beta}(a/K, b(K - 1)/K)
\]

where \( a, b \) are the hyper-parameters and the integer \( K \) the largest possible dimension for \( z_s \) (by letting \( K \to \infty \)), the length of the binary code \( z_s \), can be learned from the observed data points [39], hence, we can learn the number of the bases for each subspace using data itself. Then, we model the binary vector \( z_s \) as a random sample from the Bernoulli process parameterized by \( \Pi_s \)

\[
z_s \sim \prod_{k=1}^K \text{Ber}(z_{ks}; \pi_{ks}), \quad k = 1, \ldots, K, \ s = 1, 2, \ldots,
\]

where \( z_{ks} \) denotes the \( k \)-th element of the binary vector \( z_s \) and \( \text{Ber} \) denotes the Bernoulli distribution. To complete our probabilistic generative model, we model the weights \( \{w_{s,n}\} \), by a zero-mean Gaussian distribution with precision value \( \gamma_{s,n} \).

### 2.2. Temporally consistent clustering prior

An important problem with considering the GEM distribution as the prior distribution over the clustering indices \( \{c_n\} \) is that the Beta-distributed random weights (stick weights) \( \{\beta_s\} \) are shared among all data points, which is generally inappropriate for the time-dependent data. For example, one may wish to explicitly impose the belief that the nearby data points in the time domain are more likely to belong to the same subspace. We incorporate such prior belief into \( (2.1) \) by means of a GP on a 1-D temporal space by proposing the following temporal dependent stick weights for the GEM distribution. We call our prior model the Gaussian Process GEM (GP-GEM).

\[
p(c_n = s) = \beta_s^n, \quad \beta_s^n = \sigma(f^s(t_n)) \prod_{l=1}^{s-1} (1 - \sigma(f^l(t_n))),
\]

\[
n = 1, \ldots, N, \ s = 1, 2, \ldots \tag{2}
\]

where \( f^s(t) \sim \text{GP}(m(t), K(t, t')) \), \( f^s(t_n) \) is the value of the function \( f^s(\cdot) \) evaluated at time frame \( t_n \) of the \( n \)-th data point, and \( \sigma(\cdot) \) denotes the sigmoid function \( \sigma(x) = 1/(1 + \exp(-x)) \). The functions \( \{f^s(\cdot)\}_{s=1}^S \) are drawn from a GP with the mean function \( m(\cdot) \), which we take equal to 0 for simplicity, and the covariance function \( K(\cdot, \cdot) \). It is easy to show that the proposed prior distribution on each cluster index \( c_n \) is still a valid GEM distribution. By selecting an appropriate form of the kernel function \( K(t_i, t_j) \), which diminishes by increasing the distance between \( t_i \) and \( t_j \), the proposed GEM distribution in Eq. 2 allows for obtaining prior probabilities for the clusters that depend on the values of the temporal locations \( \{t_n\}_{n=1}^N \). Indeed, the closer the locations \( t_i \) and \( t_j \) are, the more correlated the corresponding \( f^s(t_i) \) and \( f^s(t_j) \) values should be, hence, the more similar the corresponding stick weights \( \beta_s^n \) and \( \beta_s^m \) are. Thus, the GP-GEM prior promotes, by construction, clustering of temporally adjacent data points.

The graphical representation of the proposed model is shown in Fig. 1. For computational simplicity, we truncate the GEM distribution in Eq. 2 to \( S \) term with \( \beta_s^n = 1 - \sum_{m=1}^{S-1} \beta_m^n \), with properties of this truncation discussed in [17].

### 2.3. Choice of Kernel

Since the proposed GP-GEM model is constructed using Gaussian processes there is great flexibility in the choice of covariance function (kernel). For instance, one could simply use the squared exponential (SE) function of the form \( K(t_i, t_j) = \exp(-\eta(t_i - t_j)^2) \), where \( \eta \) denotes the length scale parameter [51]. Unfortunately, using unstructured arbitrary covariance functions is costly, scaling as \( O(N^3) \) time because of the \( N \times N \) matrix inversion (see Section 3). Since \( \{t_n\} \in \mathbb{R} \) and \( t_1 < t_2 < \cdots < t_N \), we propose the covariance function \( K(t_i, t_j) = \exp(-\eta|t_i - t_j|) \), whose inverse evaluated at the data points \( \{t_n\}_{n=1}^N \) is a tridiagonal matrix, hence, \( K^{-1} \) can be computed in \( O(N) \) time [36]. Intuitively, this Ornstein-Uhlenbeck kernel induces the so-called Markovian dependence property on \( f \). More precisely, 1

\footnote{we use the notation \( K \) to denote the \( N \times N \) Gram matrix of the Gaussian process \( f \) obtained by evaluating \( K(\cdot, \cdot) \) at \( \{t_n\}_{n=1}^N \).}
the value of the function $f(t_i)$ at $t_i$ will not depend on any other point except for its immediate neighbors (according to $t_i$, $f(t_{i-1})$ and $f(t_{i+1})$, a reasonable assumption for a time-dependent data.

3. Non-parametric EM Estimation Algorithm

In this section, we develop a novel non-parametric EM algorithm for our proposed model. The approach resembles the standard EM algorithm yet still possesses the nonparametric nature in order to address the complexity of the model selection (infer both the number of subspaces and their dimensionality directly from the data). For this purpose, we consider $\{w_{s,n}\}$ and $\{\pi_s\}$ as latent (hidden) variables. By denoting $\mathcal{T} = \{t_n\}_{n=1}^N$, the goal of the EM algorithm is to maximize the following joint likelihood

$$p(X, \mathcal{T}, \{\Phi_s, \mu_s, z_s, \alpha_s, \gamma_s, c_n, f_s\}_{s=1}^S, \pi_{n}, \eta),$$

by integrating out $\{w_{s,n}, \pi_s\}$. By denoting $\Theta$ as the set of parameters and $\Omega$ as the set of latent variables, this can be accomplished by maximizing the following lower bound on the log likelihood:

$$\log p(X, \mathcal{T}, \Theta) \geq \log \mathbb{E}_{q(\Omega)}[\log p(X, \mathcal{T}, \Theta, \Omega)] - \mathbb{E}_{q(\Omega)}[\log q(\Omega)],$$

where $q(\Omega)$ is the approximate posterior distribution over the set of latent variables $\Omega$ and $\mathbb{E}_{q[\cdot]}$ denotes the expectation over the distribution $q$. For our framework to yield a computationally effective inference method, we use a factorized variational distribution

$$q(\Omega) = \prod_{s=1}^S \prod_{n=1}^N q(w_{s,n}) \prod_{k=1}^K q(\pi_{ks}),$$

where we iteratively update the posterior distribution $q(\Omega)$ in the E step and update the parameters $\Theta$ in the M step using the coordinate ascent algorithm. For simplicity, we also fix a single $K$ for all the subspaces. If $S$ and $K$ are large enough (see Section 4), the analyzed data will reveal less than $S$ subspaces and $K$ bases for each subspace, respectively.

The difficulty of applying the EM algorithm for this model lies with the logistic function in (2) which makes the update equations for $\{f_s\}_{s=1}^S$ to have non-analytic form. To address this issue, one can place an exponential lower bound on the logistic functions based on the convex duality theorem [19]. Using this theorem, a variational lower bound for the logistic sigmoid function is obtained in the form of [19]

$$\frac{1}{1 + e^{\exp(-x)}} \geq \sigma(\xi) \exp\left(\frac{(x - \xi)/2 - \lambda(\xi)(x^2 - \xi^2)}{2}\right),$$

where

$$\lambda(\xi) = \frac{-1}{2\xi} \left(\frac{1}{1 + e^{\exp(-\xi)}} - \frac{1}{2}\right).$$

and $\xi$ is the variational parameter that should be optimized to get the tightest bound. In the proposed EM algorithm, we optimize the factorized variational distribution $q(\Omega)$ in the E-step and maximizes the parameters $\Theta$ in the M-step. Detailed update equations of the proposed non-parametric EM algorithm made available in the Supplementary Material.

3.1. Missing Data

One of the main advantages of our probabilistic formulation of temporal subspace clustering is the flexibility of allowing missing data. Generalization of the proposed EM algorithm to handle missing data is straightforward and follows [10]. The only modifications come in the form of adjusted terms for data summaries. For example, in updating the precision parameters $\{\alpha_s\}$, the term $||x_n - \mu_s - \Phi_s(z_s \odot w_{s,n})||^2$ becomes

$$\sum_{\zeta \in F} (x_{n,\zeta} - \mu_s - \zeta \Phi_s(z_s \odot w_{s,n})) \times (x_{n,\zeta} - \mu_s - \zeta \Phi_s(z_s \odot w_{s,n})), $$

where $\odot$ denotes the element-wise multiplication operator, $F$ is the index set of the observed (non-missing) features for the data point $x_n$, $x_{s,\zeta}$ is the value of the present feature, $\mu_s, \zeta$ is the $\zeta$-th element of the vector $\mu_s$ and $\Phi_s, \zeta$ is the $\zeta$-th row of the matrix $\Phi_s$. Similar expressions can be derived for other data-dependent parameters.

For reconstruction purposes, given an input $x_i$ having missing features, our model computes the reconstruction of $x_i$ as

$$\hat{x}_i = \mathbb{E}_{p(x_i|\Theta)}[x_i]$$

where

$$p(x_i|\Theta) = \int \mathcal{N}(x_i; \Phi_s(z_s \odot w_{s,i}) + \mu_s, \alpha_s^{-1}I) \times q(w_{s,i}) \, dw_{s,i}$$

where $s$ is the inferred cluster index for $x_i$ ($c_i = s$), and $q(w_{s,i})$ is the variational posterior distribution of $w_{s,i}$. Since $p(x_i|\Theta)$ cannot be computed in closed form, we approximate $q(w_{s,i})$ with its mean $\mathbb{E}_{q(w_{s,i})}[w_{s,i}]$ in the above equation. Hence, $\hat{x}_i$ is computed in closed-form as

$$\hat{x}_i = \Phi_s(z_s \odot \mathbb{E}[w_{s,i}]) + \mu_s$$

4. Experimental Results

In this section, we compare our approach with several state-of-the-art subspace clustering approaches on three public human action and gesture datasets, including the Carnegie Mellon Motion Capture (Mocap) dataset, available at [http://mocap.cs.cmu.edu](http://mocap.cs.cmu.edu), Ballet Action (Ballet) dataset [http://www.humansensing.cs.cmu.edu/mad](http://www.humansensing.cs.cmu.edu/mad), and UMD Keck body-gesture (Keck) dataset
The statistics of various subjects used in the experiments are summarized in Table 1. The Mocap dataset contains 149 subjects performing several activities, from which we randomly selected 5 subjects consisting of different trials, where each trial comprises multiple activities (we selected 5 to 12 activities for each subject). Fig. 2 shows a few snapshots of some of the activities (Boxing, Drink soda, Laugh, Climb three steps) for subject 13. The statistics of various subjects used in the experiments are summarized in Table 1.

For the Mocap datasets, we are given sensor measurements at multiple joints of the human body (62 positions and joint angles) captured at different time instances. The goal is to segment the sensory data so that each cluster corresponds to the same activity. Here, each data point corresponds to a vector whose elements are the sensor measurements of different joints at a fixed time instance.

The Ballet data set contains 44 real video sequences of eight actions collected from an instructional ballet DVD [46]. Fig. 3 presents the sample frames of each action. We concatenate the randomly selected 10 sequences into a single long video sequence. The original resolution of each frame is 480 × 640. To speed up the computation, we first downsample each binary image (frame) to the size of 80 × 106. Then, we compute the Euclidean distance transform [44] as frame-level features. After that, we build a dictionary of temporal words with 100 clusters using the k-means clustering, and encode each frame as a 100 dimensional binary vector. Finally, we concatenate the 14 gesture video sequences of each subject into a single long video sequence. For comparison purposes, we contrast our proposed method (PM) with three baseline subspace clustering methods SSC [7], LRR [24], and LSR [29], and two state-of-the-art temporal subspace clustering methods OSC [40] and TSC 3 [23].

In all the experiments, we use the clustering accuracy (ACC) and normalized mutual information (NMI) as the evaluation metrics.

4.1. Hyper-Parameter Setting

For the EM algorithm, we set the truncation level for the number of subspaces and their dimension to (K = 20, S = 30) for the Mocap dataset, (K = 15, S = 10) for the Ballet dataset, and (K = 30, S = 50) for the Keck dataset. The hyper-parameters a, b of the Beta distributions are set with a = K and b = K/2 (other settings of a and b yield similar results). The parameters for the EM algorithm are initialized

<table>
<thead>
<tr>
<th></th>
<th>S-13</th>
<th>S-49</th>
<th>S-54</th>
<th>S-80</th>
<th>S-113</th>
</tr>
</thead>
<tbody>
<tr>
<td># activity</td>
<td>5</td>
<td>3</td>
<td>7</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td># instance</td>
<td>1701</td>
<td>811</td>
<td>1616</td>
<td>1877</td>
<td>2842</td>
</tr>
<tr>
<td># feature</td>
<td>62</td>
<td>62</td>
<td>62</td>
<td>62</td>
<td>62</td>
</tr>
</tbody>
</table>

Figure 2. Four activities performed by subject 13 in the Mocap dataset: Boxing, Climb three steps, Laugh, and Drink soda.

Figure 3. Sample frames from the Ballet data set. From left to right and top to bottom: Left-to-right hand opening, right-to-left hand opening, standing hand opening, leg swinging, jumping, turning, hopping, and standing still.

Figure 4. Sample gestures from the Keck dataset.

http://www.umiacs.umd.edu/zhuolin/Keckgesturedataset.html and one video scene segmentation datasets 2.

2Due to the lack of space, the video scene segmentation results are available in the Supplementary material.

Table 2. ACC with standard deviation on Subjects of Mocap dataset. The best (bold red), the second best (red).

<table>
<thead>
<tr>
<th>method</th>
<th>Subject-13</th>
<th>Subject-49</th>
<th>Subject-54</th>
<th>Subject-80</th>
<th>Subject-113</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSC</td>
<td>56.88 ± 2.01</td>
<td>85.47 ± 1.45</td>
<td>67.89 ± 2.33</td>
<td>59.04 ± 2.60</td>
<td>52.90 ± 2.27</td>
</tr>
<tr>
<td>LRR</td>
<td>57.18 ± 1.99</td>
<td>84.79 ± 1.78</td>
<td>68.81 ± 2.55</td>
<td>56.19 ± 2.50</td>
<td>53.35 ± 2.10</td>
</tr>
<tr>
<td>LSR</td>
<td>56.44 ± 2.16</td>
<td>82.27 ± 1.50</td>
<td>66.73 ± 1.90</td>
<td>67.10 ± 2.13</td>
<td>52.55 ± 2.16</td>
</tr>
<tr>
<td>OSC</td>
<td>60.66 ± 1.88</td>
<td>88.25 ± 1.70</td>
<td>70.58 ± 1.67</td>
<td>63.14 ± 2.03</td>
<td>68.79 ± 2.08</td>
</tr>
<tr>
<td>TSC</td>
<td>67.92 ± 2.00</td>
<td>95.99 ± 1.33</td>
<td>76.38 ± 2.00</td>
<td>66.33 ± 2.33</td>
<td>76.28 ± 2.10</td>
</tr>
<tr>
<td>PM (our)</td>
<td>74.75 ± 2.22</td>
<td>98.79 ± 1.88</td>
<td>81.43 ± 1.59</td>
<td>65.82 ± 2.49</td>
<td>79.50 ± 2.18</td>
</tr>
</tbody>
</table>

Table 3. NMI with standard deviation on Subjects of Mocap dataset. The best (bold red), the second best (red).

<table>
<thead>
<tr>
<th>method</th>
<th>Subject-13</th>
<th>Subject-49</th>
<th>Subject-54</th>
<th>Subject-80</th>
<th>Subject-113</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSC</td>
<td>0.5478 ± 0.023</td>
<td>0.7052 ± 0.016</td>
<td>0.6961 ± 0.028</td>
<td>0.5921 ± 0.029</td>
<td>0.6821 ± 0.035</td>
</tr>
<tr>
<td>LRR</td>
<td>0.5529 ± 0.030</td>
<td>0.6961 ± 0.020</td>
<td>0.6748 ± 0.020</td>
<td>0.6127 ± 0.032</td>
<td>0.6893 ± 0.029</td>
</tr>
<tr>
<td>LSR</td>
<td>0.5627 ± 0.027</td>
<td>0.7014 ± 0.013</td>
<td>0.6861 ± 0.018</td>
<td>0.6049 ± 0.030</td>
<td>0.6728 ± 0.033</td>
</tr>
<tr>
<td>OSC</td>
<td>0.6139 ± 0.019</td>
<td>0.8341 ± 0.017</td>
<td>0.7072 ± 0.022</td>
<td>0.6038 ± 0.025</td>
<td>0.7150 ± 0.025</td>
</tr>
<tr>
<td>TSC</td>
<td>0.6759 ± 0.020</td>
<td>0.9015 ± 0.015</td>
<td>0.7483 ± 0.024</td>
<td>0.6739 ± 0.019</td>
<td>0.7962 ± 0.029</td>
</tr>
<tr>
<td>PM (our)</td>
<td>0.7532 ± 0.024</td>
<td>0.9812 ± 0.019</td>
<td>0.8453 ± 0.024</td>
<td>0.7631 ± 0.023</td>
<td>0.8767 ± 0.023</td>
</tr>
</tbody>
</table>

Table 4. Clustering accuracies (with standard derivation) on Ballet dataset. The best (bold red), the second best (red).

<table>
<thead>
<tr>
<th>method</th>
<th>ACC</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSC</td>
<td>38.47 ± 3.07</td>
<td>0.3731 ± 0.023</td>
</tr>
<tr>
<td>LRR</td>
<td>35.15 ± 2.82</td>
<td>0.3923 ± 0.019</td>
</tr>
<tr>
<td>LSR</td>
<td>37.02 ± 3.21</td>
<td>0.4201 ± 0.020</td>
</tr>
<tr>
<td>OSC</td>
<td>41.04 ± 1.68</td>
<td>0.4008 ± 0.019</td>
</tr>
<tr>
<td>TSC</td>
<td>49.56 ± 2.31</td>
<td>0.5031 ± 0.013</td>
</tr>
<tr>
<td>PM (our)</td>
<td>53.46 ± 2.99</td>
<td>0.6206 ± 0.022</td>
</tr>
</tbody>
</table>

Table 5. Clustering accuracies (with standard derivation) on Keck dataset. The best (bold red), the second best (red).

<table>
<thead>
<tr>
<th>method</th>
<th>ACC</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSC</td>
<td>27.32 ± 3.41</td>
<td>0.3058 ± 0.038</td>
</tr>
<tr>
<td>LRR</td>
<td>15.03 ± 3.26</td>
<td>0.1159 ± 0.069</td>
</tr>
<tr>
<td>LSR</td>
<td>37.17 ± 2.52</td>
<td>0.3429 ± 0.021</td>
</tr>
<tr>
<td>OSC</td>
<td>43.02 ± 2.57</td>
<td>0.4832 ± 0.025</td>
</tr>
<tr>
<td>TSC</td>
<td>56.87 ± 2.92</td>
<td>0.6583 ± 0.020</td>
</tr>
<tr>
<td>PM (our)</td>
<td>55.49 ± 2.43</td>
<td>0.6711 ± 0.027</td>
</tr>
</tbody>
</table>

using a simple k-subspace algorithm [43]. For all the compared methods, we have tuned the parameters to get their best performance.

4.2. Results

The mean performance along with the standard deviation of each method over 5 runs on the different subjects of the three datasets is shown in Tables 2–5, from which we can infer two major points. (i) Clearly, the temporal subspace clustering methods (PM, OSC, TSC) outperform the standard subspace clustering methods (SSC, LRR, LSR) because they can exploit the temporal dependency in the data. (ii) Our proposed method also outperforms the state of the art temporal OSC and TSC methods. We attribute this
gain in performance to the probabilistic nature of the proposed method, making it more robust to outliers and noise in the datasets, in contrast to the optimization-based OSC and TSC methods.

4.3. Model Interpretation

Figs. 6 and 7 examine the number of inferred subspaces, as well as the number of bases for each subspace for the subjects 13 and 113 of the Mocap dataset. As can be seen the model infers 7 and 10 subspaces for the subjects 13 and 113, respectively. For each cluster (subspace) the subspace dimension is between 4 to 10 for the subject 13, and is between 5 to 15 for the subject 113.

In Fig. 8 we contrast the clustering performance of competing methods on Subject 54 of Mocap dataset. As can be seen, SSC, LRR and LSR can not obtain meaningful temporal segments, as they do not consider the temporal information. On the other hand, OSC, TSC and our method could obtain more coherent temporal segments. Furthermore, because of the proposed GP-GEM prior for temporal data, our model can correctly recover the subspace structures in temporal space, hence, it reveals more clear sequential subspace structures than OSC and TSC.

4.4. Missing data Experiments

To demonstrate that our method can deal with the partially observed data gracefully, we conduct experiments by considering two contexts for missing data; the case when the values in data are missing at random (MAR) and the case when the values in data are missing not-at-random (MNAR).

Since the competing methods are not designed to deal
with missing data, we use zeros to replace the missing values, which has been shown [53] to have better performance than other filling-in techniques. For the MNAR setting, we conduct experiments by removing $20 \times 20$ squares form different locations in the data matrices repeatedly until the total fraction of the missing values is no less than the pre-specified missing rate. Fig. 5 shows the clustering results (average over 5 runs) of all methods under the MAR and MNAR setting on the Subject 49 of the Mocap dataset, Ballet dataset, and Keck dataset (missing data experiments for other subjects of Mocap dataset are available in the Supplementary material). As can be seen, although the error for the MNAR case tends to be larger than in the MAR case, our probabilistic method is much more robust, particularly for the large number of missing values) than all the competing methods under different amounts of missing values.

To further investigate the reconstruction ability of our probabilistic model for missing feature scenario, we use Subject 13 of the Mocap dataset.

For this experiment, we randomly remove 5 markers (each marker corresponds to a three dimensional spatial coordinates of a human body joint) representing different body segments from each frame. Then, we recover each frame using Eq. 3.1. Since the competing methods are not suitable for recovering the missing features, we do not compare them in this experiment. Fig. 9 show the recovery results of the PM on some of the frames. The first row shows 5 randomly selected frames taken from Subject 13 of the Mocap dataset. The second row shows the same frames with 5 markers missing. Finally, the last row of the figures gives recovered frames provided by the results of our model. As can be seen, the reconstructed missing values result in poses close to the actual body postures.

5. Conclusion

In this paper, we have proposed a novel probabilistic temporal subspace clustering model by incorporating the temporal information into the model’s prior distribution that is capable of inferring the number of subspaces and their dimensions simultaneously from the available data. The temporal dependency is captured by establishing the cluster indices via a Gaussian process field followed by logistic functions. A specific kernel function is also employed to alleviate the computational issues raised by using the GPs. The experiments on three benchmark datasets demonstrate that our probabilistic method outperforms other state-of-the-art subspace clustering algorithms.

References


