DP-space: Bayesian Nonparametric Subspace Clustering with Small-variance Asymptotics

Yining Wang YININGWA@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Jun Zhu DCSZJ@TSINGHUA.EDU.CN
Dept. of Comp. Sci. & Tech., State Key Lab of Intell. Tech. & Sys., TNList, CBICR Center, Tsinghua University, China

Abstract

Subspace clustering separates data points approximately lying on union of affine subspaces into several clusters. This paper presents a novel nonparametric Bayesian subspace clustering model that infers both the number of subspaces and the dimension of each subspace from the observed data. Though the posterior inference is hard, our model leads to a very efficient deterministic algorithm, *DP-space*, which retains the nonparametric ability under a small-variance asymptotic analysis. DP-space monotonically minimizes an intuitive objective with an explicit tradeoff between data fitness and model complexity. Experimental results demonstrate that DP-space outperforms various competitors in terms of clustering accuracy and at the same time it is highly efficient.

1. Introduction

Given a collection of data points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n \in \mathbb{R}^D$, subspace clustering is the task that groups the data points into K components by finding K affine subspaces of dimensions $d_1, \cdots, d_K < D$ such that each data point lies in (or near) a particular subspace. Subspace clustering has found many applications in computer vision, including motion segmentation (Vidal et al., 2008) and face clustering (Ho et al., 2003). See (Vidal, 2010) for an excellent review.

Many subspace clustering methods have been developed. One particularly interesting example is the mixture of probabilistic PCAs (MPPCA) (Tipping & Bishop, 1999), which represents each affine subspace by a probabilistic PCA (pPCA) model and posits that each data point comes from a mixture of pPCA models. Compared to alternative algebraic and geometry solutions (Ma et al., 2007; Fischler & Bolles, 1981; Elhamifar & Vidal, 2013), the MPPCA formulation enjoys the advantage of being more tolerate to noisy and outlier values. Model parameters of MPPCA could be learned via Expectation-Maximization (EM) (Tipping & Bishop, 1999). However, a major drawback of MPPCA is that both the number of clusters K and the dimension of each subspace d_k must be specified a priori (Vidal, 2010). This causes difficult model selection problems, because unlike PCA, we need to select both K and d_k at the same time. When K or D is large, model selection soon becomes intractable due to an exponential number of subspace dimension configurations.

In this paper, we present Dirichlet process mixture of PCAs (DP-PCA), a novel nonparametric Bayesian model for subspace clustering that automatically resolves model complexity from data. One challenge of DP-PCA is on defining proper priors that take the nonparametric requirements of clustering as well as unknown subspace dimensions into consideration. For clustering, we adopt the commonly used Chinese restaurant process (CRP (Aldous, 1985; Pitman, 1995), a marginalization of Dirichlet process) prior on cluster assignment variables. In addition, we propose a novel hierarchical prior to facilitate a nonparametric treatment on subspace dimensions, which is significantly different from the mixture of factor analysis (MFA) (Chen et al., 2010) model. More importantly, though the posterior inference is hard, our carefully designed model leads to an efficient deterministic algorithm, named *DP-space*, which resembles the EM algorithm for MPPCA but still possesses the nonparametric nature of DP-PCA to resolve model complexity. DP-space is provable to converge by monotonically minimizing an intuitive objective during iterations, and empirically it achieves increased efficiency and clustering accuracy on synthetic and real-world datasets.

Technically, we first derive a partially collapsed Gibbs (PCG) sampler (van Dyk & Park, 2008) for the DP-PCA model. The PCG sampler is elegant in theory but unfortunately impractical in computation. By carefully performing the small-variance asymptotics (SVA) analysis (Kulis & Jordan, 2012), our PCG sampler leads to the DP-space

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

algorithm. Compared to existing SVA analysis for many popular nonparametric Bayesian models (Kulis & Jordan, 2012; Jiang et al., 2012; Broderick et al., 2013; Roychowdhury et al., 2013; Wang & Zhu, 2014), our work makes novel contributions by presenting a new nonparametric model as well as a non-trivial generalization of existing SVA methods for the challenging task of subspace clustering. Furthermore, we note that there are subspace clustering methods that do not require full knowledge of K and d_k. We discuss such methods in Sec. 5 and show that they are less accurate, not as efficient as DP-space, or cannot handle both unknown K and d_k settings.

2. Preliminaries

We briefly review some preliminary knowledge and present a proposition that will be used later.

2.1. Projection onto subspaces

A d-dimensional affine subspace $S \subseteq \mathbb{R}^D, d < D$ can be expressed as

$$S = S(\mathbf{W}, \boldsymbol{\mu}) = \{\boldsymbol{x} \in \mathbb{R}^D : \boldsymbol{x} = \mathbf{W}\boldsymbol{y} + \boldsymbol{\mu}, \boldsymbol{y} \in \mathbb{R}^d\}, \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{D\times d}$ is a subspace projection matrix and $\boldsymbol{\mu} \in \mathbb{R}^D$ specifies the offset of S. Without loss of generality, we assume $\mathbf{W}$ has full column rank. By singular value decomposition we know that there exist $l'_1 \geq \cdots \geq l'_d > 0$, $\mathbf{U}_d \in \mathbb{R}^{D\times d}$ with orthonormal columns and an orthogonal matrix $\mathbf{R} \in \mathbb{R}^{d\times d}$ such that

$$\mathbf{W} = \mathbf{U}_d \mathbf{diag}(l'_1, \cdots, l'_d)\mathbf{R}^\top. \tag{2}$$

For a vector $\boldsymbol{x} \in \mathbb{R}^D$ and a subspace $S \subseteq \mathbb{R}^D$, define the distance between $\boldsymbol{x}$ and S as

$$d(\boldsymbol{x}, S) := \inf_{\boldsymbol{y}\in S} \|\boldsymbol{x} - \boldsymbol{y}\|_2. \tag{3}$$

Proposition 1 provides an easy way to compute $d(\boldsymbol{x}, S)$. We defer the proof to Appendix A.

Proposition 1. *Fix a d-dimensional affine subspace $S(\mathbf{W}, \boldsymbol{\mu})$ and let $\mathbf{U} = [\mathbf{U}_d, \mathbf{U}_{D-d}] \in \mathbb{R}^{D\times D}$ be an orthogonal matrix associated with $\mathbf{W}$. Here $\mathbf{U}_d$ is defined in Eq. (2) and $\mathbf{U}_{D-d} \in \mathbb{R}^{D\times(D-d)}$ can be any orthonormal basis complementing $\mathbf{U}_d$. Then for all $\boldsymbol{x} \in \mathbb{R}^D$, we have*

$$d(\boldsymbol{x}, S)^2 = \sum_{j=d+1}^{D} \left[\mathbf{U}^\top(\boldsymbol{x} - \boldsymbol{\mu})\right]_j^2. \tag{4}$$

One consequence of Proposition 1 is that $d(\cdot, S(\mathbf{W}, \boldsymbol{\mu}))$ does not depend on the eigenvalues $l'_1, \cdots, l'_d$ of $\mathbf{W}$. We shall use this important property in the small-variance asymptotic analysis later.

2.2. Mixtures of probabilistic PCA models

A mixture of probabilistic PCA (MPPCA) model (Tipping & Bishop, 1999) $\mathcal{M}$ posits that a data point is generated via a mixture of K subspace clusters. Mathematically, it can be represented as $\mathcal{M} = (\boldsymbol{\pi}, \{\mathbf{W}_k, \boldsymbol{\mu}_k, \sigma_k^2\}_{k=1}^K)$ where $\mathbf{W}_k \in \mathbb{R}^{D\times d_k}$ and $\boldsymbol{\mu}_k \in \mathbb{R}^D$ specify the subspace for the kth cluster, σ_k^2 indicates the variance and $\boldsymbol{\pi}$ is a mixture probability distribution over the K clusters. Let $z_i \in [K]$ denote the cluster assignment of data $\boldsymbol{x}_i$ and define $\mathbf{W} = \{\mathbf{W}_k\}$, $\boldsymbol{\mu} = \{\boldsymbol{\mu}_k\}$. Then, the likelihood of $\boldsymbol{x}_i$ is:

$$p(\boldsymbol{x}_i|\mathcal{M}) = \sum_{k=1}^{K} p(z_i = k)p(\boldsymbol{x}_i|z_i = k, \mathbf{W}, \boldsymbol{\mu}), \tag{5}$$

where $p(z_i = k) = \pi_k$ and the per-subspace likelihood model (conditioned on $\mathbf{W}$ and $\boldsymbol{\mu}$) is defined as

$$\begin{aligned} p(\boldsymbol{x}_i|z_i = k) &= \int_{\mathcal{R}^{d_k}} \mathcal{N}(\boldsymbol{x}_i; \mathbf{W}_k\boldsymbol{y}_i + \boldsymbol{\mu}_k, \sigma_k^2\mathbf{I})\mathrm{d}p_0(\boldsymbol{y}_i) \\ &= \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \mathbf{W}_k\mathbf{W}_k^\top + \sigma_k^2\mathbf{I}); \end{aligned} \tag{6}$$

here we assume the prior $p_0(\boldsymbol{y}_k)$ is a standard Gaussian.

Given the number of subspaces K, the dimensions of subspaces $\{d_k\}_{k=1}^K$ and a collection of data points $\{\boldsymbol{x}_i\}_{i=1}^n$, an EM procedure (Tipping & Bishop, 1999) can be used to learn the model parameters $\{\mathbf{W}_k, \boldsymbol{\mu}_k, \sigma_k^2\}_{k=1}^K$. $\boldsymbol{\pi}$ could also be inferred from learnt parameters.

3. Dirichlet Process Mixtures of PCAs

We now present Dirichlet process mixtures of PCAs (DP-PCA), a novel nonparametric Bayesian subspace clustering model that could handle an arbitrary number of subspaces. Furthermore, the dimension of each subspace does not need to be known *a priori* either.

3.1. The DP-PCA Model

DP-PCA is a Bayesian mixture model that adopts the same mixture model as in Eq. (5), but with several changes. First, for simplicity, we assume the same variance parameter σ^2 is shared among all subspaces and serves as a hyperparameter. This assumption greatly simplifies the small-variance analysis and it does not cause performance deterioration, as shown in the experiments.

Second, we need to specify a prior distribution $p_0(\boldsymbol{z}, \mathbf{W}, \boldsymbol{\mu})$ where $\boldsymbol{z} = \{z_i\}_{i=1}^n$. To simplify derivation, we impose an improper non-informative prior $\mathcal{N}(\mathbf{0}, \rho^2\mathbf{I}_{D\times D})$ with $\rho^2 \to \infty$ on subspace offsets $\boldsymbol{\mu}_k$. The prior on the cluster assignment variables $\boldsymbol{z}$ is a Chinese Restaurant Process (CRP) (Aldous, 1985; Pitman, 1995) in order to facilitate a nonparametric treatment of clusters.

More specifically, denoting $\boldsymbol{z}_{-i}$ as $\{z_j\}_{j\neq i}$, we have

$$p_0(z_i = k|\boldsymbol{z}_{-i}) = \begin{cases} n_{-i,k}/Z, \text{if } n_{-i,k} > 0; \\ \alpha/Z, \quad \text{otherwise.} \end{cases} \tag{7}$$

where $n_{-i,k} = \sum_{j\neq i} \delta^k_{z_j}$, $\delta^k_z = I[z_j = k]$ and Z is a normalization constant. α is a hyper-parameter of CRP.

The prior distribution on $\mathbf{W}_k$ is much more delicate, due to two reasons: 1) we do not have easy conjugate priors for $\mathbf{W}_k$, and 2) we need to facilitate a nonparametric treatment on the dimension of each $\mathbf{W}_k$. Our solution is a hierarchical prior:

$$p_0(\mathbf{W}_k) = \sum_{d_k=0}^{D-1} p_0(\mathbf{W}_k|d_k)p_0(d_k). \tag{8}$$

First, once the dimension d_k of $\mathbf{W}_k$ is given, we define the prior $p_0(\mathbf{W}_k|d_k)$ as follows.[1] Using the strategy in (Minka, 2000; Zhang et al., 2004), we can express $\mathbf{W}_k$ as

$$\mathbf{W}_k = \mathbf{U}^{(k)}_{d_k}(\mathbf{L}^{(k)}_{d_k} - \sigma^2\mathbf{I}_{d_k\times d_k})^{1/2}\mathbf{R}^{(k)}_{d_k}, \tag{9}$$

where $\mathbf{U}^{(k)}_{d_k} \in \mathbb{R}^{D\times d_k}$ has orthonormal columns, $\mathbf{R}^{(k)}_{d_k} \in \mathbb{R}^{d_k\times d_k}$ is an orthogonal matrix and $\mathbf{L}^{(k)}_{d_k} = \mathbf{diag}(l^{(k)}_1, \cdots, l^{(k)}_{d_k})$. Note that Eq. (9) is identical to Eq. (2) if define $l^{(k)}_j = l'_j + \sigma^2$. Hence, there exists an orthogonal matrix $\mathbf{U}^{(k)} = \left[\mathbf{U}^{(k)}_{d_k}, \mathbf{U}^{(k)}_{D-d_k}\right] \in \mathbb{R}^{D\times D}$ such that

$$\begin{aligned}\boldsymbol{\Sigma}_k &:= \mathbf{W}_k\mathbf{W}_k^\top + \sigma^2\mathbf{I}_{D\times D} \\ &= \mathbf{U}^{(k)}\mathbf{diag}(l^{(k)}_1, \cdots, l^{(k)}_{d_k}, \sigma^2, \cdots, \sigma^2)\mathbf{U}^{(k)\top}.\end{aligned} \tag{10}$$

To simplify notations, we will omit the superscript (k) in the sequel when the meanings of the cluster assignments are clear from the context.

To ensure identifiability, we assume following (Zhang et al., 2004) that $l_1 > l_2 > \cdots > l_{d_k} > \sigma^2$, and hence the prior distribution over $l_1, \cdots, l_{d_k}$ can be written as

$$p_0(l_1, \cdots, l_{d_k}|\sigma^2) = d_k! \prod_{j=1}^{d_k} p_0(l_j)\cdot I[l_1 > \cdots > l_{d_k} > \sigma^2],$$

and the prior on l_j^{-1} is assumed to be a Gamma distribution with hyper-parameters a and b:

$$p_0(l_j|a, b) = \Gamma(l_j^{-1}; a, b) = \frac{b^a}{\Gamma(a)}(l_j^{-1})^{a-1}e^{-bl_j^{-1}}. \tag{11}$$

Finally, we assume an improper non-informative prior on the orthonormal matrix $\mathbf{U}$.

[1] If the dimension or rank of $\mathbf{W}_k$ does not agree with d then $p_0(\mathbf{W}_k|d)$ should be zero. In addition, if $d_k = 0$ then $\mathbf{W}_k = \mathbf{0}$ with probability 1.

For the prior on the dimension, we simply use a truncated geometric distribution:

$$p_0(d_k) = \frac{1-r}{1-r^D}\cdot r^{d_k}, \quad \forall d_k = 0, 1, \cdots, D-1, \tag{12}$$

where $r \in (0, 1)$ is a hyper-parameter. This prior formulates the intuition that we favor smaller subspace dimensions because smaller dimension means lower model complexity. In the sequel, we use $\boldsymbol{d} = (d_1, \cdots, d_K)$ to denote all subspace dimensions.

3.2. A Partially Collapsed Gibbs Sampler

Let $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ denote the training data. We need to infer the posterior distribution of variables $\mathbf{W}, \boldsymbol{d}, \boldsymbol{\mu}, \boldsymbol{z}$:

$$\begin{aligned}p(\mathbf{W}, \boldsymbol{d}, \boldsymbol{\mu}, \boldsymbol{z}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})}p_0(\boldsymbol{z})\prod_{k=1}^{K^+}\Big(p_0(\boldsymbol{\mu}_k)p_0(d_k) \\ \cdot p_0(\mathbf{W}_k|d_k)\Big)\prod_{i=1}^n p(\boldsymbol{x}_i|z_i, \mathbf{W}_k, \boldsymbol{\mu}_k),\end{aligned} \tag{13}$$

where K^+ is the number of active subspaces and we drop the hyper-parameters $(\alpha, r, \rho, a, b, \sigma)$ for notational simplicity. Unsurprisingly, the posterior is intractable. Furthermore, a standard Gibbs sampler is not applicable either because we cannot sample d_k assuming the other variables (e.g., $\mathbf{W}_k$) are fixed, since by definition this will give us $d_k = \text{rank}(\mathbf{W}_k)$ almost surely, which is not acceptable.

Below, we present a partially collapsed Gibbs (PCG) sampler (van Dyk & Park, 2008) which circumvents the above-mentioned challenges for an ordinary Gibbs sampler. PCG sampling often improves the convergence by replacing some of the conditional distributions of an ordinary Gibbs sampler with some marginal distributions. The sampler iteratively performs the following steps:

Update of $\boldsymbol{d}$: This is the collapsed step of our PCG sampler. Specifically, we sample d_k from its conditional distribution with $\mathbf{W}_k$ integrated out; that is,

$$\begin{aligned}p(d_k|\mathbf{X}, \boldsymbol{z}, \boldsymbol{\mu}, a, b, r) &\propto p(\mathbf{X}|d_k, \boldsymbol{z}, \boldsymbol{\mu}, a, b)p_0(d_k|r) \\ &= p_0(d_k|r)\cdot\int p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \boldsymbol{z})\mathrm{d}p_0(\mathbf{W}|d_k, a, b).\end{aligned} \tag{14}$$

Update of $\boldsymbol{\mu}$: Because both the prior and likelihood of $\boldsymbol{\mu}_k$ are Gaussian, the posterior distribution of $\boldsymbol{\mu}_k$ is Gaussian, too. Furthermore, since we impose a non-informative prior on $\boldsymbol{\mu}_k$, its conditional distribution can be simplified as

$$p(\boldsymbol{\mu}_k|\mathbf{W}, \boldsymbol{z}, \boldsymbol{X}) = \mathcal{N}(\boldsymbol{\mu}_k; \bar{\boldsymbol{x}}, \boldsymbol{\Sigma}_k), \tag{15}$$

where $\bar{\boldsymbol{x}} = \frac{1}{n_k}\sum_{i=1}^n 1_{[z_i=k]}\cdot\boldsymbol{x}_i$, with n_k the number of instances assigned to the kth cluster, and $\boldsymbol{\Sigma}_k$ is defined in

Eq. (10). Intuitively, the mean of the posterior distribution $\bar{\boldsymbol{x}}$ is the sample mean of all data points in cluster k.

Update of $\boldsymbol{z}$: We need to consider two cases. First, for an existing component k, we have

$$p(z_i = k|\boldsymbol{z}_{-i}, \mathbf{W}_k, \boldsymbol{\mu}_k, \boldsymbol{x}_i) \propto n_{-i,k} \cdot \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),$$

where $n_{-i,k}$ is the number of data instances other than i that are assigned to the k-th cluster. Second, for a new component k_{new}, we have

$$p(z_i = k_{new}|\alpha, \boldsymbol{x}_i) \propto \alpha \cdot \int p(\boldsymbol{x}_i|\mathbf{W}, \boldsymbol{\mu}) \mathrm{d}p_0(\mathbf{W}, \boldsymbol{\mu}). \quad (16)$$

Update of $\mathbf{W}$: Once $\boldsymbol{z}$ is given, we can sample each $\mathbf{W}_k$ separately. Specifically, when the dimension d_k is fixed, we can sample the matrix $\mathbf{W}_k$ by sampling its associated orthonormal matrix $\mathbf{U}$ and $l_1, \cdots, l_{d_k}$ as in Eq. (10). Let

$$\hat{\mathbf{S}} = \frac{1}{n_k} \sum_{i,j=1}^{n} \delta_{z_i}^k \delta_{z_j}^k (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_j - \boldsymbol{\mu}_k)^\top \quad (17)$$

be the sample covariance matrix and $\hat{\mathbf{S}} = \mathbf{AGA}^\top$ be its eigen decomposition. Here $\delta_{z_i}^k$ is an indicator variable that equals 1 if $z_i = k$ and 0 otherwise. Because a non-informative prior is imposed on $\mathbf{U}$, we can take $\mathbf{U}$ to be the maximum likelihood estimation [2] , which is the eigen-matrix $\mathbf{A}$ of the covariance matrix $\hat{\mathbf{S}}$ (Zhang et al., 2004). After $\mathbf{U}$ is determined, we use Gibbs sampling to sample $l_1, \cdots, l_{d_k}$ separately from their conditional distribution (Zhang et al., 2004): (assuming $l_{d_k+1} = \sigma^2$ and $l_0 = +\infty$)

$$p(l_j|\boldsymbol{l}_{-j}, \mathbf{X}, \boldsymbol{z}, a, b) = \Gamma\left(l_j^{-1}; \frac{D}{2} + a, \left(\frac{D\mathbf{G}_{jj}}{2} + b\right)^{-1}\right) \cdot I[l_{j+1} < l_j < l_{j-1}],$$

4. Small-variance asymptotic analysis

Though the PCG sampler is elegant, the integrals in (14) and (16) are unfortunately very hard to evaluate due to non-conjugacy. Below, we develop an efficient algorithm with provable convergence guarantees by performing small-variance asymptotic (SVA) analysis to both the PCG sampler and the target posterior.

4.1. SVA behavior of the PCG sampler

We first analyze the behavior of the PCG sampler when the variance of the likelihood model goes to zero. Specifically, we consider the likelihood model with scaled variance parameter $\tilde{\sigma}^2 = \sigma^2/\beta$, where $\beta > 0$ is a scaling constant. Intuitively, once we take the limit $\beta \to \infty$ the posterior variance diminishes. Similarly, we need to scale the hyper-parameter b as $\tilde{b} = b/\beta$ so that the variance of the prior distribution of l^{-1} goes to zero. For the prior distribution over d, we scale its hyper-parameter r using a different scaling constant β' as $\tilde{r} = r\exp(-\beta')$ and establish a connection between β and β' as $\beta' = s/\delta^2 \cdot \beta$ for some constant s.

Update of $\boldsymbol{\mu}$: Under the scaling $\tilde{\sigma}^2 = \sigma^2/\beta$, we have

$$p(\boldsymbol{\mu}_k|\mathbf{W}, \boldsymbol{z}, \mathbf{X}, \beta) = \mathcal{N}(\boldsymbol{\mu}_k; \bar{\boldsymbol{x}}, \mathbf{W}_k\mathbf{W}_k^\top + \sigma^2/\beta \cdot \mathbf{I}). \quad (18)$$

Recall that $\boldsymbol{\Sigma}_k = \mathbf{W}_k\mathbf{W}_k^\top + \sigma^2\mathbf{I}$. Suppose $\boldsymbol{\Sigma}_k = \mathbf{U}^{(k)}\mathbf{diag}(l_1, \cdots, l_{d_k}, \sigma^2, \cdots, \sigma^2)\mathbf{U}^{(k)^\top}$. As $\beta \to \infty$, the update rule becomes

$$\boldsymbol{\mu}_k = \bar{\boldsymbol{x}} + \mathbf{U}^{(k)}\boldsymbol{w}, \quad (19)$$

where $w_j \sim \mathcal{N}(0, l_j)$ for $j \le d_k$ and $w_j = 0$ for $j > d_k$. Alternatively, we could simply take $\boldsymbol{\mu}_k = \bar{\boldsymbol{x}}$ if a deterministic update rule is desired.

Update of $\boldsymbol{z}$: Define $\boldsymbol{v}_{i,k} = \mathbf{U}^{(k)^\top}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)$. Then the likelihood $\mathcal{N}(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ can be rewritten as $\mathcal{N}(\boldsymbol{v}_{i,k}; \mathbf{0}, \mathbf{diag}(l_1, \cdots, l_{d_k}, \sigma^2, \cdots, \sigma^2))$. Consequently, as $\beta \to \infty$, we have (for existing k)

$$\begin{aligned} & p(z_i = k|\boldsymbol{z}_{-i}, \boldsymbol{x}_i, \boldsymbol{\mu}_k, \mathbf{W}_k) \\ \to & \frac{1}{Z} \cdot \exp\left(-\beta/\sigma^2 \cdot \sum_{j=d_k+1}^{D} [\boldsymbol{v}_{i,k}]_j^2 + o(\beta)\right) \\ = & \frac{1}{Z} \exp\left(-\frac{\beta \cdot d(\boldsymbol{x}_i, S_k)^2}{\sigma^2} + o(\beta)\right), \end{aligned} \quad (20)$$

where $S_k = S(\mathbf{W}_k, \boldsymbol{\mu}_k)$ denotes the kth subspace and the last equality holds due to Proposition 1. Through a more involved analysis (see Appendix B.1), we derive a similar limiting result for the probability of creating a new cluster:

$$p(z_i = k_{new}|\alpha, \boldsymbol{x}_i) \to \frac{\alpha}{Z}\exp(o(\beta)). \quad (21)$$

Finally, scale the hyper-parameter α as $\alpha = \exp(-\beta\lambda/\sigma^2)$ and define a cost function $Q_i(k)$ as

$$Q_i(k) := \begin{cases} d(\boldsymbol{x}_i, S_k)^2, & n_{-i,k} > 0; \\ \lambda. & \text{otherwise.} \end{cases} \quad (22)$$

By Eq. (20) and (21), when $\beta \to \infty$ the posterior distribution of z_i will concentrate on the cluster k^* with the smallest cost $Q_i(k^*)$. In other words, we could deterministically update z_i using

$$z_i' = \underset{k=1:K^+ +1}{\operatorname{argmin}} Q_i(k). \quad (23)$$

Intuitively, the cost $Q_i(k)$ measures how close the given data point $\boldsymbol{x}_i$ is to an existing subspace S_k, and Eq. (23)

[2] This is in fact an approximation of the PCG sampler. Nevertheless, in Section 4 we show that the MLE corresponds to the update rule after applying small-variance asymptotic analysis.

says we should always put $\boldsymbol{x}_i$ into the cluster whose subspace is the closest to $\boldsymbol{x}_i$. Furthermore, if $\boldsymbol{x}_i$ is far away from any existing subspaces, a new cluster is created with only one data point, $\boldsymbol{x}_i$. This type of updates can be seen in a number of recent development of fast inference algorithms derived using SVA analysis (Kulis & Jordan, 2012; Jiang et al., 2012; Wang & Zhu, 2014).

Update of $\mathbf{W}$: When d_k is fixed and $\tilde{b} \to 0$, the update of $\mathbf{U}^{(k)}$ remains unchanged while the update of $l_j^{(k)}$ becomes

$$l_j^{(k)} = \frac{\mathbf{G}_{jj}}{1 + 2(a-1)/D}. \tag{24}$$

With $\mathbf{U}^{(k)}$ and $\boldsymbol{l}^{(k)}$, we calculate $\mathbf{W}_k$ as:

$$\mathbf{W}_k = \mathbf{U}_{d_k}^{(k)} \mathbf{diag}\left(\sqrt{l_1^{(k)}}, \cdots, \sqrt{l_{d_k}^{(k)}}\right), \tag{25}$$

where $\mathbf{U}_{d_k}^{(k)} \in \mathbb{R}^{D \times d_k}$ is the left d_k columns of $\mathbf{U}^{(k)}$.

Update of $\boldsymbol{d}$: Under the SVA setting, the conditional distribution of d_k is

$$p(d_k|\mathbf{X}, \boldsymbol{z}, \boldsymbol{\mu}, a, b, r, \beta, \beta') = \frac{1}{Z(\mathbf{X})} p_0(d_k|r, \beta')$$
$$\int p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \boldsymbol{z}, \sigma^2, \beta) \mathrm{d}p_0(\mathbf{W}|d_k, a, b, \beta) =: q(d_k|\beta, \beta').$$

Through a careful limiting analysis presented in Appendix B.2, we can show that [3]

$$\lim_{\beta\to\infty} q(d_k|\beta, \beta') = \frac{1}{Z} \exp\left(-\beta \cdot \left(\frac{s d_k}{\sigma^2} + \inf_{\mathbf{W}\in\mathbb{R}^{D\times d_k}} \sum_{i=1}^n \delta_{z_i}^k \frac{d(\boldsymbol{x}_i, S(\mathbf{W}, \boldsymbol{\mu}_k))^2}{\sigma^2}\right) + o(\beta)\right). \tag{26}$$

Here hyper-parameter s is defined as $\beta' = s/\delta^2 \cdot \beta$, which works as a parameter of connection between β and β'.

Let $\mathbf{U}$ be an orthogonal matrix associated with $\mathbf{W}$, as defined in Eq. (10). As we have mentioned in previous sections, when the offset $\boldsymbol{\mu}_k$ is fixed the distance $d(\boldsymbol{x}_i, S(\mathbf{W}, \boldsymbol{\mu}_k))$ only depends on the projection matrix $\mathbf{U}$. By PCA, for a fixed d, the optimal $\mathbf{U}$ can be obtained as $\mathbf{U} = \mathbf{A}_d$ where $\mathbf{A}_d$ is the matrix of top d eigenvectors of the $\hat{\mathbf{S}}$. By Eq. (26), as $\beta \to \infty$ the posterior distribution of d_k will concentrate on one specific value and subsequently we get a deterministic update rule for d_k:

$$d_k = \underset{d=0:D-1}{\operatorname{argmin}} \left\{ s \cdot d + \sum_{i=1}^n \delta_{z_i}^k \cdot d(\boldsymbol{x}_i, S(\mathbf{A}_d, \boldsymbol{\mu}_k))^2 \right\}. \tag{27}$$

Intuitively, the update rule (27) strikes for a balance between model complexity and data fitness as measured by the term $\sum_i \delta_{z_i}^k \cdot d(\boldsymbol{x}_i, S(\mathbf{A}_d, \boldsymbol{\mu}_k))^2$. When the subspace dimension d increases the distance decreases and hence the model better fits the training data. It contrasts the first term $s \cdot d$, which controls the model complexity by imposing a linear penalty on subspace dimension d.

[3] $\lim_{\beta\to\infty} f(\beta) = g(\beta)$ means $\lim_{\beta\to\infty} f(\beta)/g(\beta) = 1$.

Algorithm 1 The DP-space algorithm

Input: data $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^n$, parameters a, λ, s.
Initialize: $K^+ = 1$ cluster, with $d_1 = 0$, $\boldsymbol{\mu}_1 = \bar{\mathbf{X}}$.
while not converge **do**
  Update $\boldsymbol{\mu}$: $\boldsymbol{\mu}_k = \frac{1}{n_k} \sum_{i=1}^n \delta_{z_i}^k \cdot \boldsymbol{x}_i$, $k \in [K^+]$.
  for each $k \in [K^+]$ **do**
    Compute $\hat{\mathbf{S}}_k$ using Eq. (17);
    Compute the diagonalization $\hat{\mathbf{S}}_k = \mathbf{AGA}^\top$.
    Update $\mathbf{U}^{(k)}$: $\mathbf{U}^{(k)} = \mathbf{A}$.
    Update d_k using Eq. (27)
    Update $l_j^{(k)}$ using Eq. (24), $j \in [d_k]$;
    Update $\mathbf{W}_k$ using Eq. (25);
  end for
  Update $\boldsymbol{z}$: set each z_i using Eq. (23);
  Remove empty clusters and re-calculate K^+.
end while
Output: K^+, $\{\mathbf{W}_k, d_k, \boldsymbol{\mu}_k\}_{k=1}^{K^+}$ and $\{z_i\}_{i=1}^n$.

4.2. SVA of the posterior

We now derive a deterministic loss function defined on the training data set via SVA analysis on the posterior distribution and shows that Algorithm 1 converges by decreasing the loss function at each iteration. To begin with, we write the CRP prior as an exchange partition probability function (Aldous, 1985; Pitman, 1995):

$$p_0(\boldsymbol{z}|\alpha) = \alpha^{K^+-1} \frac{\Gamma(\alpha+1)}{\Gamma(\alpha+n)} \prod_{k=1}^{K^+} (n_k - 1)!, \tag{28}$$

where K^+ is the number of non-empty clusters and n_k is the number of instances in cluster k. Subsequently, the posterior of DP-PCA can be expressed as:

$$p(\boldsymbol{z}, \mathbf{W}, \boldsymbol{d}, \boldsymbol{\mu}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \alpha^{K^+} \psi(\boldsymbol{n}) \prod_{k=1}^{K^+} \Big(p_0(\boldsymbol{\mu}_k|\rho) p_0(d_k|r) \cdot p_0(\mathbf{W}_k|d_k, a, b) \Big) \prod_{i=1}^n p(\boldsymbol{x}_i|z_i = k, \mathbf{W}_k, \boldsymbol{\mu}_k). \tag{29}$$

Here $\psi(\boldsymbol{n}) = \prod_{k=1}^{K^+} (n_k - 1)!$. $\frac{\Gamma(\alpha+1)}{\alpha\Gamma(\alpha+n)}$ is absorbed in $Z(\mathbf{X})$ because it does not depend on model parameters. With $a = 1$ and the scaling $\tilde{\sigma}^2 = \sigma^2/\beta$, $\tilde{r} = r \exp(-s/\delta^2 \cdot \beta)$, $\alpha = \exp(-\lambda/\delta^2 \cdot \beta)$, $\tilde{b} = b/\beta$, we have

$$p(\boldsymbol{z}, \mathbf{W}, \boldsymbol{d}, \boldsymbol{\mu}|\mathbf{X}, \beta) = 1/Z(\mathbf{X}, \beta) \exp\left(-\beta/\sigma^2 \left(\lambda K^+\right.\right.$$

$$+s\sum_{k=1}^{K^+} d_k + \sum_{i=1}^{n} d(\boldsymbol{x}_i, S_{z_i})^2\Bigg) + o(\beta)\Bigg). \quad (30)$$

As a result, when we take $\beta \to \infty$ the posterior distribution will concentrate on its mode, that is, the model $\boldsymbol{\theta}^* = (\boldsymbol{z}^*, \mathbf{W}^*, \boldsymbol{d}^*, \boldsymbol{\mu}^*)$ that minimizes the loss

$$\mathcal{L}(\boldsymbol{z}, \mathbf{W}, \boldsymbol{d}, \boldsymbol{\mu}) := \lambda K^+ + s\sum_{k=1}^{K^+} d_k + \sum_{i=1}^{n} d(\boldsymbol{x}_i, S_{z_i})^2. \quad (31)$$

The loss function has an intuitive tradeoff between data fitness measured by the last term and model complexity measured by $(K^+, \{d_k\})$. It also reveals practical interpretations of the two hyper-parameters λ and s. For example, λ could be viewed as the distance threshold between a data point and the subspace it is associated with. similarly, the parameter s characterizes the residue after PCA for each subspace and the algorithm is encouraged to increase d_k once the residue exceeds s. In some applications (e.g. clustering gene sequences) hyper-parameters could be directly set according to their physical meanings.

Theorem 1 states that the DP-space algorithm monotonically decreases this loss function at each iteration. We defer its proof to Appendix B.3.

Theorem 1. *Let $\mathbf{W}^{(t)}, \boldsymbol{d}^{(t)}$, $\boldsymbol{\mu}^{(t)}$ and $\boldsymbol{z}^{(t)}$ be model parameters output by Algorithm 1 after iteration t. Then for all t we have $\mathcal{L}(\mathbf{W}^{(t+1)}, \boldsymbol{d}^{(t+1)}, \boldsymbol{\mu}^{(t+1)}, \boldsymbol{z}^{(t+1)}) \leq \mathcal{L}(\boldsymbol{W}^{(t)}, \boldsymbol{d}^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{z}^{(t)})$.*

5. Related work

The Agglomerative Lossy Compression (ALC) (Ma et al., 2007) algorithm builds a clustering solution by first creating one cluster for each data point, and then combining existing clusters in a greedy manner. The algorithm does not need prior knowledge of the number of clusters and subspace dimensions. Instead, a distortion parameter δ is required to reflect the distortion level in the underlying data. ALC is different from DP-space in two ways: first, ALC minimizes the coding length needed to fit data points instead of finding solutions near MLE; second, unlike ALC, DP-space first puts all instances into a giant cluster and tries to create new clusters in the process.

The Random Sample Consensus (RANSAC) (Fischler & Bolles, 1981) algorithm fits a subspace of dimension d by randomly sampling $d+1$ points from the training data. It does not need to know the number of subspaces. However, the algorithm does require the knowledge of subspace dimensions, and a drawback of RANSAC is that it generally performs bad when there are many subspaces or the subspace dimensions are incorrectly set (Yang et al., 2006).

The Sparse Subspace Clustering (SSC) (Elhamifar & Vidal, 2013) algorithm solves a LASSO regression problem for each data point to form a similarity graph. Then spectral clustering methods are used to cluster data into subspaces. SSC performs well on real-world data sets (Vidal, 2010) and in general K and d_k do not need to be known a priori. However, SSC is based on ℓ_1 optimization techniques, which could be slow on large data sets. Our experiments show that DP-space is much faster than SSC while achieving slightly worse performance.

(Chen et al., 2010) proposed a nonparametric Bayesian mixture of factor analysis (MFA). Similar to ours, both the mixture number and subspace rank can be inferred from data. However, the nonparametric treatments of subspace dimension are different. In (Chen et al., 2010) an auxiliary indicator vector z is introduced with a Beta process prior while in our model we place a geometric prior directly on the subspace rank. Our method is arguably more direct, though it results in non-conjugacy. Furthermore, by SVA analysis we obtain a new efficient algorithm.

Finally, small-variance asymptotic (SVA) analysis is a powerful method that speeds up the inference of nonparametric Bayesian models, with many recent advances on performing SVA analysis to the popular nonparametric models, including Dirichlet process mixture models (Kulis & Jordan, 2012; Jiang et al., 2012), nonparametric latent factor models (Broderick et al., 2013), infinite HMMs (Roychowdhury et al., 2013) and infinite SVMs (Wang & Zhu, 2014). Our work contributes by presenting a novel nonparametric Bayesian model as well as a non-trivial extension of existing SVA methods to the challenging task of subspace clustering.

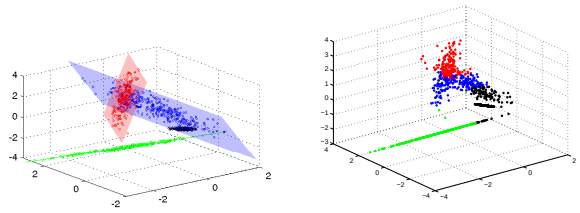
6. Experiments

We compare DP-space with several competitors on both synthetic and real-world datasets. All methods are implemented in Matlab. Throughout the experiments we always set the hyper-parameter a to be 1, under which no shrinkage on l_j is imposed. The other hyper-parameters are selected via cross-validation.

6.1. Synthetic examples

We first analyze how DP-space uses subspaces of different dimensions to cluster synthetic data. The dataset contains $10,000$ data points from $\mathbb{R}^3$. They are divided into 4 clusters with equal probability. Among the 4 clusters, two lie on 1-d affine subspaces and the other two lie on 2-d affine subspaces. A Gaussian white noise $\mathcal{N}(\mathbf{0}, 0.05\mathbf{I}_{D\times D})$ is imposed on each data point. We show the clustering results of DP-space under different settings of hyper-parameters, and also compare it with K-means and the ground truth.

Fig. 1 shows the ground truth (with subspaces depicted in



(a) (b)

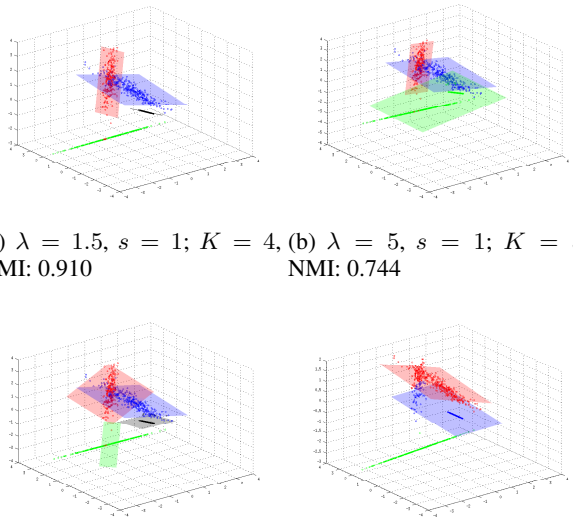
Figure 1. (a) the ground truth; and (b) the clustering result of K-means with $K = 4$.

different colors) and the clustering results of K-means with $K = 4$. As shown in Figure 1(b), K-means does not capture the concept of subspace well. For instance, the last few instances lying on the green line are misclassified because they are closer to the center of another cluster in Euclidean distance. Consequently, the Normalized Mutual Information (NMI) [4] score is as low as 0.610, much worse than DP-space shown in Figure 2.

Fig. 2 shows the clustering results on the synthetic dataset by DP-space with different hyper-parameter settings. When λ and s are set properly as in Fig. 2(a) the clustering result is very close to the ground truth. Both the number of clusters K and subspace dimensions d_k are inferred correctly. When λ increases as in Figure 2(b), we expect to get a smaller number of clusters because we place a larger penalty on K^+. In particular, the two 1-d subspaces are merged into a single 2-d subspace cluster. On the other hand, smaller s would result in subspaces of higher dimension. This is shown in Fig. 2(c): we still get 4 clusters, but now all subspaces are of dimension two. Furthermore, s also controls the number of clusters in an indirect way because for any additional cluster we add a penalty on its subspace dimension. For instance, in Fig. 2(d) the λ parameter remains 1.5 but s is as large as 10.0. We can see that again the number of clusters returned decreases to 3.

We next turn to larger synthetic datasets. The dataset now contains $100,000$ data points from $\mathbb{R}^{10}$. Data points are divided with equal probability into $K = 6$ clusters. The underlying subspace dimensions of the K clusters are set as $\boldsymbol{d} = (d_1, \cdots, d_6) = (2, 2, 3, 3, 4, 4)$. A similar Gaussian white noise is imposed on each data point. We report the clustering performance and running time for DP-space and other baseline algorithms on 10 i.i.d. generated synthetic datasets. Similar to (Kulis & Jordan, 2012), we use NMI to evaluate clustering performance with different cluster numbers. The hyper-parameters of non-parametric

[4] Let $\mathcal{A} = \{a_1, \cdots, a_n\}$, $\mathcal{B} = \{b_1, \cdots, b_m\}$ be the output and ground truth clusterings. The NMI score is defined as $\text{NMI}(\mathcal{A}, \mathcal{B}) := \frac{I(\mathcal{A};\mathcal{B})}{[H(\mathcal{A})+H(\mathcal{B})]/2}$, where $I(\mathcal{A};\mathcal{B}) = \sum_{j,k} \frac{|\omega_j \cap c_k|}{N} \log \frac{N|\omega_j \cap c_k|}{|\omega_j||c_k|}$ and $H(\mathcal{A}) = -\sum_j \frac{|\omega_j|}{N} \log \frac{|\omega_j|}{N}$. N is the total number of examples.



(a) $\lambda = 1.5$, $s = 1$; $K = 4$, NMI: 0.910 (b) $\lambda = 5$, $s = 1$; $K = 3$, NMI: 0.744

(c) $\lambda = 1.5$, $s = 0.5$; $K = 4$, NMI: 0.898 (d) $\lambda = 1.5$, $s = 10$; $K = 3$, NMI: 0.727

Figure 2. Clustering results of DP-space under different settings of λ and s, where K is the number of recovered clusters.

Table 1. Average NMI, running time (seconds), cluster number and subspace dimension on synthetic datasets. For parametric models the number of clusters and subspace dimensions are fixed.

Algorithm	NMI	Time	Cluster no.	Dim.
K-means	.713	2.6	6	-
DP-means	.697	23.6	26.3	-
MPPCA, $d = 2$	.847	43.6	6	2
MPPCA, $d = 4$	.924	32.6	5	4
MPPCA, $d = 4$	.999	19.8	6	4
DP-space	.972	12.5	6.3	4.2

models are selected on one tenth of the data set using NMI. We can see that DP-space achieves similar NMI scores with EM-MPPCA while being faster. Furthermore, DP-space correctly selects the number of clusters and subspace dimensions. On the other hand, when K or d_k is incorrectly specified, the performance of EM-MPPCA deteriorates.

6.2. Application to Motion Segmentation

Motion segmentation usually refers to the task of separating the movements of multiple rigid-body objects from video sequences. Subspace clustering methods are popular in this task by simultaneously clustering the point trajectory data, which can be extracted using tracking methods, into multiple subspaces and finding a low-dimensional subspace fitting each group of points. See (Vidal, 2010; Elhamifar & Vidal, 2013) for a comprehensive treatment.

In Table 2, we compare the performance of several subspace clustering algorithms on the Hopkins 155 motion

Table 2. Average classification error (%) and running time (seconds) on the Hopkins-155 dataset, where "-" means the algorithm usually doesn't operate under the specific projection setting.

	$r = 5$		$r = 4N/SP$	
	error	time	error	time
GPCA	10.34	13.2	11.55	18.2
RANSAC	9.76	1.4	-	-
ALC	3.76	347.8	3.37	352.1
EM-MPPCA-a	18.56	3.8	24.88	6.2
EM-MPPCA-m	3.49	3.8	7.28	6.2
DP-space	3.32	2.1	3.29	2.1
LRR	-	-	3.74	294.5
SSC-ADMM	-	-	2.41	542.3

segmentation dataset (Tron & Vidal, 2007). Each sequence in the dataset contains motions of $N = 2$ or 3 different objects. A detailed summary of statistics for the Hopkins-155 dataset can be found in Appendix C.1. Performance is measured in terms of classification error, which is the percentage of misclassified video sequences. Specifically, we consider all permutations of cluster assignments and compare them against the ground-truth classification. If more clusters than the number of objects are returned, all video sequences in extra clusters will be ruled as misclassified.

We compare the performance of DP-space with EM-MPPCA and several other baseline algorithms, including Generalized PCA (GPCA) (Vidal et al., 2005), RANSAC (Fischler & Bolles, 1981), ALC (Ma et al., 2007), LRR (Liu et al., 2013) [5] and SSC (Elhamifar & Vidal, 2013). Segmentation performance is measured in terms of classification error, which is the percentage of misclassified point trajectories. Following the convention in (Vidal, 2010), we first project point projectories onto an r-dimensional subspace using PCA, with r varying from 5 to $4N$ or a level that preserves sparsity (SP). [6] We run the EM-MPPCA algorithm using 10 random initializations on each video sequence and report both the average performance (MPPCA-a) and the best performance (MPPCA-m) over different initializations. Parameters of the DP-space algorithm are selected using 30% of the ground-truth labels according to NMI. Values of λ range from 10^{-3} to 10^2 and values of s range from 10^{-2} to 10^2. Classification errors for GPCA, RANSAC, ALC are cited from (Vidal, 2010) and from (Elhamifar & Vidal, 2013) for LRR and SSC. Running time for LRR is cited from (Liu et al., 2013) and running time for SSC is measured using implementations provided in (hop). For SSC we use the ADMM implementation because it is faster and also more accurate than the original CVX implementation (Elhamifar & Vidal, 2013). For all algorithms the reported running time does not include the time for parameter selection or cross-validation.

[5] For LRR a heuristic post-processing step is adopted, as implemented in the code (lrr).

[6] This requires $r \geq 8 \log(2F/r)$, where F is the number of frames collected.

Table 3. Clustering error and running time for SSC-ADMM under constrained number of ADMM iterations (T).

T	5	10	20	30	40
error	5.03	3.64	3.33	3.20	2.88
time	2.8	5.6	10.9	16.0	21.4

From Table 2 we see that DP-space performs far better than GPCA, RANSAC and the EM-MPPCA algorithm, and its performance is comparable to ALC and LRR. Although DP-space does not perform as good as SSC in terms of classification accuracy, it is significantly faster than SSC because SSC is an optimization based algorithm and requires solving P Lasso problems, each with P variables (where P is the number of points in each video sequence). For the Hopkins 155 dataset P ranges from 100 to 400, as shown in Appendix C.1. In Appendix C.2 we present a more detailed comparison of classification error.

Since optimization based methods like SSC-ADMM are slower but more accurate, we further investigate the trade-off between clustering accuracy and running time. In Table 3 we constrain the maximum number of ADMM iterations for SSC and report the corresponding clustering accuracy and running time. The results show that when SSC-ADMM is run with very few iterations its performance suffers. On the other hand, SSC-ADMM takes much longer to achieve the same level of clustering accuracy as DP-space does.

6.3. Discussion

Unlike DP-means, the DP-space algorithm always achieves a significant improvement over the parametric MPPCA model on real-world datasets, as shown in Table 2. There could be several reasons for this phenomenon.

First, DP-space enjoys the advantage of selecting different subspace dimension for different clusters, while the dimension is assumed to be the same for all components in MPPCA, for which an exhaustive search over all possible combinations of subspace dimension is simply intractable. This provides additional modeling flexibility for DP-space.

Second, the EM-MPPCA algorithm is very sensitive to its initialization (Elhamifar & Vidal, 2013). This fact is verified in Table 2: when the best performance of EM-MPPCA over 10 different initializations is reported, the classification accuracy is much better than when only the average performance is reported, and in fact the performance is comparable with DP-space and ALC.

Acknowledgements

This work was conceived when Y.W. was at Tsinghua. The work is supported by the National 973 Basic Research Program of China (No. 2013CB329403, 2012CB316301), National NSF of China (No. 61322308, 61332007), and Tsinghua Initiative Scientific Research Program (No. 20121088071, 20141080934).

References

Code for several subspace clustering algorithms. http://vision.jhu.edu/code/. [Online; accessed 16-Aug-2014].

Motion segmentation and face clustering by lrr. https://sites.google.com/site/guangcanliu/. [Online; accessed 17-Jan-2015].

Aldous, D. Exchangeability and related topics. *École d'Été de Probabilités de Saint-Flour XIII1983*, pp. 1–198, 1985.

Broderick, Tamara, Kulis, Brian, and Jordan, Michael. Mad-bayes: Map-based asymptotic derivations from bayes. In *ICML*, 2013.

Chen, Minhua, Silva, Jorge, Paisley, John, Wang, Chunping, Dunson, David, and Carin, Lawrence. Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds. *IEEE Transactions on Signal Processing*, 58(12):6140–6155, 2010.

Elhamifar, Ehsen and Vidal, Rene. Sparse subspace clustering: Algorithm, theory and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (11):2765–2781, 2013.

Fischler, Martin and Bolles, Robert. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

Ho, Jeffrey, Yang, Ming-Hsuan, Lim, Jongwoo, Lee, Kuang-Chih, and Kriegman, David. Clustering appearances of objects under varying illumination conditions. In *CVPR*, 2003.

Jiang, Ke, Kulis, Brian, and Jordan, Michael. Small-variance asymptotics for exponential family dirichlet process mixture models. In *NIPS*, 2012.

Kulis, Brian and Jordan, Michael. Revisiting k-means: New algorithms via bayesian nonparametrics. In *ICML*, 2012.

Liu, Guangcan, Lin, Zhouchen, Yan, Shuicheng, Sun, Ju, Yu, Yong, and Ma, Yi. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (1):171–184, 2013.

Ma, Yi, Derksen, Harm, Hong, Wei, and Wright, John. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1546–1562, 2007.

Minka, Thomas. Automatic choice of dimensionality for pca. In *NIPS*, 2000.

Pitman, J. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102(2):145–158, 1995.

Roychowdhury, Anirban, Jiang, Ke, and Kulis, Brian. Small-variance asymptotics for hidden markov models. In *NIPS*, 2013.

Tipping, Michael and Bishop, Christopher. Mixtures of probabilistic principle component analyzers. *Neural Computation*, 11(2):443–482, 1999.

Tron, Roberto and Vidal, Rene. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, 2007.

van Dyk, David and Park, Taeyoung. Partially collapsed gibbs samplers: Theory and methods. *Journal of the American Statistical Association*, 103(482):790–796, 2008.

Vidal, Rene. A tutorial on subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2010.

Vidal, Rene, Ma, Yi, and Sastry, Shankar. Generalized principal component analysis (gpca). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12): 1945–1959, 2005.

Vidal, Rene, Tron, Roberto, and Hartley, Richard. Multiframe motion segmentation with missing data using power factorization and gpca. *International Journal of Computer Vision*, 79(1):85–105, 2008.

Wang, Yining and Zhu, Jun. Small variance asymptotics for dirichlet process mixtures of svms. In *AAAI*, 2014.

Yang, Allen, Rao, Shankar, and Ma, Yi. Robust statistical estimation and segmentation of multiple subspaces. In *CVPR Workshop*, 2006.

Zhang, Zhihua, Chan, Kap-Luk, Kwok, James, and Yeung, Dit-Yan. Bayesian inference on principle component analysis using reversible jump markov chain monte carlo. In *AAAI*, 2004.

A. Proof of Proposition 1

Proof. We only prove the proposition for $\boldsymbol{\mu} = 0$. If $\boldsymbol{\mu} \neq 0$, we could simply take the mapping $\boldsymbol{x} \to \boldsymbol{x} - \boldsymbol{\mu}$, $\boldsymbol{y} \to \boldsymbol{y} - \boldsymbol{\mu}$ and complete the proof in a similar manner.

When $\mathbf{W}$ is full rank, it is well known that the projected vector $\boldsymbol{y} \in S \subseteq \mathbb{R}^D$ has the following form:

$$\boldsymbol{y} = \mathbf{W}(\mathbf{W}^\top \mathbf{W})^{-1}\mathbf{W}^\top \boldsymbol{x} = (\mathbf{U}_d \mathbf{U}_d^\top)\boldsymbol{x}.$$

Next, note that

$$\mathbf{U}_d \mathbf{U}_d^\top = \mathbf{U}\mathbf{diag}(\mathbf{I}_d, \mathbf{O})\mathbf{U}^\top. \tag{32}$$

Therefore,

$$\begin{aligned}
\|\boldsymbol{x} - \boldsymbol{y}\|^2 =& \|\boldsymbol{x} - (\mathbf{U}_d \mathbf{U}_d^\top)\boldsymbol{x}\|^2 \\
=& \|\boldsymbol{x} - \mathbf{U}\mathbf{diag}(1, \cdots, 1, 0, \cdots, 0)\mathbf{U}^\top \boldsymbol{x}\|^2 \\
=& \|\mathbf{U}^\top \boldsymbol{x} - \mathbf{diag}(1, \cdots, 1, 0, \cdots, 0)\mathbf{U}^\top \boldsymbol{x}\|^2 \\
=& \sum_{j=d+1}^{D} [\mathbf{U}^\top \boldsymbol{x}]_j^2.
\end{aligned}$$

□

B. Technical details of SVA analysis

B.1. Derivation of the update rule for z

For creating a new cluster, we use Laplace approximation to approximate the integration. We first write the conditional distribution as

$$p(z_i = k_{new} | \mathbf{X}, \rho, a, b, r, \alpha) = \frac{\alpha}{Z} \sum_{d=1}^{D} p_0(d|r) \int p(\boldsymbol{x}_i | \mathbf{W}, \boldsymbol{\mu}, \sigma^2) \mathrm{d}p_0(\mathbf{W}, \boldsymbol{\mu} | d, \rho, a, b) =: \frac{1}{Z} \sum_{d=1}^{D} p_0(d|r) J_d. \tag{33}$$

Subsequently, the scaled conditional distribution can be written as

$$p(z_i = k_{new} | \mathbf{X}, \rho, a, b, r, \alpha, \beta) = \frac{\alpha}{Z} \sum_{d=1}^{D} p_0(d|r, \beta') J_d(\beta), \tag{34}$$

where

$$J_d(\beta) = \int p(\boldsymbol{x}_i | \mathbf{W}, \boldsymbol{\mu}, \sigma^2, \beta) \mathrm{d}p_0(\mathbf{W}, \boldsymbol{\mu} | d, \rho, a, b, \beta). \tag{35}$$

Define $\boldsymbol{\theta}_d := (\mathbf{W}, \boldsymbol{\mu})$ with $\mathbf{W} \in \mathbb{R}^{D \times d}$ and

$$f_{d,\beta}(\boldsymbol{\theta}_d) := \beta^{-1} \cdot p(\boldsymbol{x}_i | \mathbf{W}, \boldsymbol{\mu}, \sigma^2, \beta) p_0(\mathbf{W}, \boldsymbol{\mu} | d, a, b, \rho, \beta). \tag{36}$$

Using Laplace's approximation, we have (as $\beta \to \infty$)

$$J_d(\beta) = \int \exp(-\beta f_{d,\beta}(\boldsymbol{\theta}_d)) \mathrm{d}\boldsymbol{\theta}_d = \frac{\exp(-\beta f_{d,\beta}(\hat{\boldsymbol{\theta}}_d))}{(2\pi/\beta)^{-D(d+1)/2}} \left(\left| \frac{\partial^2 f_{d,\beta}(\hat{\boldsymbol{\theta}}_d)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} \right|^{-1/2} + o(1) \right), \tag{37}$$

where $\hat{\boldsymbol{\theta}}_d = \mathrm{argmin}_{\boldsymbol{\theta}_d} f_{d,\beta}(\boldsymbol{\theta}_d)$. Note that [7]

$$\lim_{\beta \to \infty} f_{d,\beta}(\boldsymbol{\theta}_d) = \exp\left(-\sigma^{-2} \cdot \sum_{j=d+1}^{D} [\mathbf{U}^\top (\boldsymbol{x}_i - \boldsymbol{\mu})]_j^2 \right) = \exp\left(-\frac{d(\boldsymbol{x}_i, S)^2}{\sigma^2} \right). \tag{38}$$

[7] Recall that $\lim_{x \to \infty} f(x) = g(x)$ means $\lim_{x \to \infty} \frac{f(x)}{g(x)} = 1$.

As a result, $f_{d,\beta}(\hat{\boldsymbol{\theta}}_d) = 0$ (taking $\boldsymbol{\mu} = \boldsymbol{x}_i$) and

$$\lim_{\beta\to\infty} J_d(\beta) = (2\pi/\beta)^{D(d+1)/2} \cdot g_d(\boldsymbol{x}_i), \tag{39}$$

where $g_d(\boldsymbol{x}_i)$ only depends on d and $\boldsymbol{x}_i$. Therefore,

$$\lim_{\beta\to\infty} p(z_i = k_{new}) = \lim_{\beta\to\infty} \frac{\alpha}{Z} \sum_{d=0}^{D} \exp(-\beta' d + o(\beta)) = \frac{\alpha}{Z} \exp(o(\beta)). \tag{40}$$

B.2. Derivation of the update rule for W_k

Define

$$K_{d_k}(\beta) := \int p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \boldsymbol{z}, \sigma^2, \beta) p_0(\mathbf{W}|d_k, a, b, \beta) \mathrm{d}\mathbf{W}. \tag{41}$$

Then the (scaled) posterior distribution of d_k can be written as

$$p(d_k|\mathbf{X}, \boldsymbol{z}, \boldsymbol{\mu}, a, b, r, \beta) = \frac{1}{Z(\mathbf{X})} p_0(d_k|r, \beta') K_{d_k}(\beta). \tag{42}$$

Next, define

$$F_{d_k,\beta}(\mathbf{W}) := \beta^{-1} p(\mathbf{X}|\mathbf{W}, \boldsymbol{\mu}, \boldsymbol{z}, \sigma^2, \beta) p_0(\mathbf{W}|d_k, a, b, \beta). \tag{43}$$

Using Laplace approximation, we have

$$K_{d_k}(\beta) = \int \exp(-\beta F_{d_k,\beta}(\mathbf{W})) \mathrm{d}\mathbf{W} = \frac{\exp(-\beta F_{d_k,\beta}(\hat{\mathbf{W}}))}{(2\pi/\beta)^{-Dd_k/2}} \left(\left| \frac{\partial^2 F_{d_k,\beta}(\hat{\mathbf{W}})}{\partial \mathbf{W} \partial \mathbf{W}^\top} \right|^{-1/2} + o(1) \right), \tag{44}$$

where $\hat{\mathbf{W}}$ is the minimizer of $F_{d_k,\beta}(\cdot)$. Note that for any full-rank $\mathbf{W} \in \mathbb{R}^{D\times d_k}$,

$$\lim_{\beta\to\infty} F_{d_k,\beta}(\mathbf{W}) = \exp\left(-\sum_{i=1}^{n} 1_{[z_i=k]} \sum_{j=d_k+1}^{D} \frac{[\mathbf{U}^\top(\boldsymbol{x}_i - \boldsymbol{\mu}_k)]_j^2}{\sigma^2} - \sum_{j=1}^{d_k} \frac{l_j^{-1}}{b} \right). \tag{45}$$

Taking $l_j \to \infty$, it is then clear that

$$\lim_{\beta\to\infty} F_{d_k,\beta}(\hat{\mathbf{W}}) = \exp\left(-\inf_{\mathbf{U}} \sum_{i=1}^{n} 1_{[z_i=k]} \sum_{j=d_k+1}^{D} \frac{[\mathbf{U}^\top(\boldsymbol{x}_i - \boldsymbol{\mu}_k)]_j^2}{\sigma^2} \right) \tag{46}$$

$$= \exp\left(-\inf_{\mathbf{W}\in\mathbb{R}^{D\times d_k}} \sum_{i=1}^{n} 1_{[z_i=k]} \cdot \frac{d(\boldsymbol{x}_i, S(\mathbf{W}, \boldsymbol{\mu}_k))^2}{\sigma^2} \right). \tag{47}$$

Here the second equation is due to the fact that $d(\cdot, S(\mathbf{W}, \boldsymbol{\mu}))$ does not depend on eigenvalues of $\mathbf{W}$, and hence optimization over $\mathbf{U}$ is equivalent to optimization over $\mathbf{W}$.

B.3. Proof of Theorem 1

Proof. We first prove that for each cluster $k \in [K]$, after updating the subspace projection matrix $\mathbf{W}_k$ (along with its dimension d_k) and the offset $\boldsymbol{\mu}_k$, the loss function $\mathcal{L}$ does not increase. When subspace dimension $d_k = d$ is fixed, the update rule

$$\boldsymbol{\mu}_k = \frac{1}{n_k} \sum_{z_i=k} \boldsymbol{x}_i, \quad \mathbf{U}_{d_k}^{(k)} = \mathbf{A}_d \tag{48}$$

is exactly the same with principle component analysis (PCA) for the top d_k principle directions.. As a result, the subspace S_k given by $S(\mathbf{W}_k, \boldsymbol{\mu}_k)$ minimizes the total squared distance of data points and S_k within the k-th cluster (i.e.,

$\sum_{z_i=k} d(\boldsymbol{x}_i, S_k)^2$). Note again that the distance $d(\boldsymbol{x}_i, S(\boldsymbol{W}_k, \boldsymbol{\mu}_k))$ only depends on $\boldsymbol{\mu}_k$ and the orthogonal matrix $\mathbf{U}_d^{(k)}$ associated with $\mathbf{W}_k$. The eigenvalues of $\mathbf{W}_k$ do not affect the distance.

We have proved that given $d_k = d$, the update rule given in Eq. (48) chooses $\mathbf{W}_k$ and $\boldsymbol{\mu}_k$ that minimizes the total squared distance for each instance. The update rule for d_k given in Eq. (27) shows that we want to select the dimension d that minimizes the sum of total squared distance and a linear penalty term $s \cdot d$. This is consistent with the deterministic loss function $\mathcal{L}$ shown in Eq. (31). So after updates of $\mathbf{W}$, $\boldsymbol{d}$ and $\boldsymbol{\mu}$ the loss function does not increase.

Next, we turn to the update of cluster assignments $\boldsymbol{z}$. We want to prove that after each update of z_i for some data point $\boldsymbol{x}_i$ the loss function does not increase. This part of analysis resembles the analysis of K-means and DP-means algorithm (Kulis & Jordan, 2012). When we assign z_i to an existing cluster it is clear the distance $d(\boldsymbol{x}_i, S_k)$ does not increase and neither does the total loss. When z_i is assigned to a new cluster, we lose a $d(\boldsymbol{x}_i, S_k)$ cost and gains a λ cost because of creating a new cluster. This does not increase the total loss function $\mathcal{L}$, however, by the definition of $Q_i(k)$ and the update rule of z_i shown in Eq. (23). Note that the new cluster will have a dimension of zero, so no extra penalty term is incurred.

□

C. Details of Hopkins-155 experiments

C.1. Some statistics of the Hopkins-155 dataset

Table 4 gives some statistics of the Hopkins-155 dataset, including the number of sequences (n), the number of points (P) and the number of frames (F) per sequence. In Table 4 the notation *Check-2* refers to all checker board video sequences that contain 2 motions.

Table 4. Some statistics of the Hopkins 155 dataset (Tron & Vidal, 2007)

Dataset	n	P	F
Check-2	78	291	28
Check-3	26	437	28
Traffic-2	31	241	30
Traffic-3	7	332	31
Articul.-1	11	155	40
Articul.-2	2	122	31
All	155	vary	vary

C.2. Detailed performance comparison

In Table 5 we provide detailed performance comparison for the DP-space algorithm and its competitors, including both the mean and median classification error on each of the video sequence groups. Note that the results for EM-MPPCA-m are only included for reference because they are not directly comparable with other performance results.

Table 5. Classification error (%) of several algorithms on the Hopkins 155 dataset

	Check-2		Check-3		Traffic-2		Traffic-3		Articul.-2		Articul.-3		All	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
GPCA (5)	6.09	1.03	31.95	32.93	1.41	**0.00**	19.83	19.55	2.88	**0.00**	16.85	16.85	10.34	2.54
GPCA (4N)	4.78	0.51	36.99	36.26	1.63	**0.00**	39.68	40.92	6.18	3.20	29.62	29.62	11.55	1.36
RANSAC (5)	6.52	1.75	25.78	26.00	2.55	0.21	12.83	11.45	7.25	2.64	21.38	21.38	9.76	3.21
ALC (5)	2.56	**0.00**	6.78	0.92	2.83	0.30	**4.01**	1.35	6.90	0.89	7.25	7.25	3.76	**0.26**
ALC (SP)	**1.49**	0.27	**5.00**	**0.66**	1.75	1.51	8.86	**0.51**	10.70	0.95	21.08	21.08	3.37	0.49
EM-MPPCA (5,a)	18.13	17.48	29.07	30.10	12.84	13.32	18.98	20.32	13.54	15.21	23.49	23.49	18.56	17.56
EM-MPPCA (4N,a)	24.85	24.75	37.01	38.07	18.46	18.15	29.03	26.04	12.90	14.11	32.11	32.11	24.88	23.44
DP-space (5)	2.13	0.48	9.86	7.26	**0.53**	0.20	4.31	2.57	3.79	1.90	**1.75**	**1.75**	**3.32**	0.53
DP-space (4N)	2.08	0.38	8.77	3.94	**1.33**	0.78	7.01	7.27	**2.07**	0.43	16.95	16.95	**3.29**	0.57
EM-MPPCA (5,m)	2.95	0.00	10.76	10.37	0.52	0.00	1.96	0.99	0.46	0.00	9.33	9.33	3.49	0.00
EM-MPPCA (4N,m)	6.56	3.55	19.35	19.96	0.81	0.00	12.03	9.39	0.18	0.00	16.14	16.14	7.28	1.09