TRANSLATED POISSON MIXTURE MODEL FOR STRATIFICATION LEARNING

By

Gloria Haro

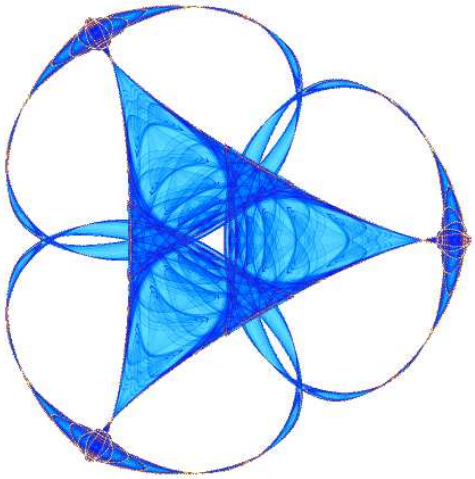
Gregory Randal

and

Guillermo Sapiro

IMA Preprint Series # 2174

(September 2007)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA

400 Lind Hall

207 Church Street S.E.

Minneapolis, Minnesota 55455–0436

Phone: 612/624-6066 Fax: 612/626-7370

URL: http://www.ima.umn.edu

Translated Poisson Mixture Model for Stratification Learning

Gloria Haro
Dept. Teoria del Senyal i Comunicacions
Universitat Politècnica de Catalunya, Spain
gloria@gps.tsc.upc.edu

Gregory Randall
Instituto de Ingeniería Eléctrica
Universidad de la República, Uruguay
randall@fing.edu.uy

Guillermo Sapiro
Dept. of Electrical and Computer Engineering
University of Minnesota, USA
guille@umn.edu

Abstract

A framework for the regularized and robust estimation of non-uniform dimensionality and density in high dimensional noisy data is introduced in this work. This leads to learning stratifications, that is, mixture of manifolds representing different characteristics and complexities in the data set. The basic idea relies on modeling the high dimensional sample points as a process of Translated Poisson mixtures, with regularizing restrictions, leading to a model which includes the presence of noise. Theoretical asymptotic results for the model are presented as well. The presentation of the theoretical framework is complemented with artificial and real examples showing the importance of regularized stratification learning in high dimensional data analysis in general and computer vision and image analysis in particular.

1 Introduction

Recently, there has been significant interest in analyzing the intrinsic structure of high dimensional data, this is commonly known as *manifold learning*, e.g., [4, 6, 9, 20, 23, 27, 32]. Often, points that live in a high dimensional space can be parametrized by a number of parameters much smaller than the ambient dimension. A representation (embedding) of the data in a lower dimensional space is very helpful for analysis and computations on the dataset.

Most of the works on manifold learning rely on the hypothesis that all the points under analysis are samples of the same manifold and thus there is a unique intrinsic dimension. However, this is often not a correct assumption. It is likely that, for example, a collection of image portraits of the same person under varying pose and illumination, lies on a manifold defined by a set of parameters related to the variations in pose and illumination. On the other hand, let us consider a set of images representing scanned digits. It might happen that the images representing the digit '1' can be described with a different number of parameters than the images for the digit '2.' Videos of diverse human motions contain the same complexity variability. In these cases, it is important to detect that there are different complexities present in the same (noisy) point cloud data. This is the subject of this work.

This problem, clustering-by-dimensionality and *stratification learning*, has recently been explored in a handful of works. Barbará and Chen, [3], proposed a hard clustering technique based on the fractal dimension (box-counting). Starting from an initial clustering, they incrementally add points into the cluster for which the change in the fractal dimension after adding the point is the lowest. They also find the number of clusters and the intrinsic dimension of the underlying manifolds. Gionis *et al.*, [13], propose a two-step algorithm: First, they estimate the local correlation dimension and density for each point; then, standard clustering techniques are used to cluster the two-dimensional representation (dimension + density) of the data. Souvenir and Pless, [30], use an Expectation Maximization (EM) type of technique, combined with weighted geodesic multidimensional scaling (weighted ISOMAP [32]). The weights measure how well each point fits the underlying manifold defined by the current set of points in the cluster. After clustering, each cluster dimensionality is estimated following [23]. Vidal *et al.*, [18, 34], cluster linear subspaces with an algebraic geometric method based on polynomial differentiation, called Generalized PCA (GPCA), which also finds the number of linear subspaces and their intrinsic dimensions. Goh and Vidal [14] extend [26] to cluster a union of J, non-intersecting, k-connected nonlinear manifolds. It is done with the vectors spanning the null space of the LLE matrix [27], which are a linear combination of the membership vectors and the embedding vectors of the J connected components. The work of Mordohai and Medioni, [25], es-

timates the local dimension using tensor voting. Cao and Haralick, [7], propose a hard clustering by dimensionality: First, local dimensionality is computed via local PCA; and then, neighboring points are clustered together if they have the same dimension and if the error of representing the new cluster as a combination of basis functions in a kernel-based feature space is small. Among these clustering-by-dimensionality techniques, only the one by Cao and Haralick includes spatial information in order to obtain a regularized classification. Recently, Lu and Vidal, [24], combined GPCA with an additional spatial constraint in a k-means fashion. They showed that, by adding this constraint, the classification is improved in the intersection of the linear subspaces. From the computational geometry perspective, a Voronoi-based technique to compute local dimensionality has been introduced in [11], and demonstrated for 3D point cloud data. The diffusion distance framework, [8, 22], can work with stratifications, though no explicit estimation of the clusters is performed and single maps into Euclidean space are performed for the whole data set. Recently, and following in part the theory of persistent topology [12], a framework for studying stratas based on local homology has been introduced in [5].

These recent works have clearly shown the necessity to go beyond manifold learning, into "stratification learning." In our work, we do not assume linear subspaces, and we simultaneously estimate the soft clustering and the intrinsic dimension and density of the clusters while being robust to noise and outliers. This collection of attributes is not shared by any of the pioneering works just described. Our approach is an extension of the Levina and Bickel's local dimension estimator [23]. They proposed to compute the intrinsic dimension at each point using a Maximum Likelihood (ML) estimator based on a Poisson distribution. We propose to compute a ML on the whole point cloud data at the same time (and not one for each point independently), based on a Translated Poisson mixture model, which models the presence of noise and permits to have different classes (each one with their own dimension and sampling density). This technique automatically gives a soft clustering according to dimensionality and density, with an estimation of both quantities for each class. A preliminary version of this work was presented in [15] and a regularized version together with asymptotic results in [16]. These techniques are particular cases of the more general Translated Poisson model introduced in this paper in order to handle noise.

The remainder of this paper is organized as follows: In Section 2 we review the method proposed by Levina and Bickel, [23], which gives a local estimation of the intrinsic dimension and has inspired our work. We reformulate this approach in Section 3 in order to include the presence of noise in the statistical model. Section 4 explains our approach for robust stratification learning. We show experiments with synthetic and real data in Section 5, including comparisons with critical literature, and finally, conclusions are presented in Section 6.

2 Local intrinsic dimension estimation

Levina and Bickel, [23], proposed a geometric and probabilistic method which estimates the local dimension and density of a point cloud data. This dimension estimator is equivalent to the one proposed in [31] in the context of dynamical systems. Their approach is based on the idea that if we sample an m-dimensional manifold with T points, the proportion of points that fall into a ball around a point x_t is $\frac{k}{T} \approx \rho(x_t)V(m)R_k(x_t)^m$. The given point cloud, embedded in high dimensions D, is $X = \{x_t \in \mathbb{R}^D; t = 1, \ldots, T\}$, k is the number of points inside the ball, $\rho(x_t)$ is the local sampling density at point x_t, $V(m)$ is the volume of the unit sphere in $\mathbb{R}^m$, and $R_k(x_t)$ is the Euclidean distance from x_t to its k-th nearest neighbor (kNN). Then, they consider the inhomogeneous process $N(R, x_t)$, which counts the number of points falling into a small D-dimensional sphere $B(R, x_t)$ of radius R centered at x_t. This is a binomial process, and some assumptions need to be done to proceed. First, if $T \to \infty$, $k \to \infty$, and $k/T \to 0$, then we can approximate the binomial process by a Poisson process. Second, the density $\rho(x_t)$ is considered constant inside the sphere, a valid assumption for small R. With these assumptions, the rate λ of the counting process $N(R, x_t)$ can be written as

$$\lambda(R, x_t) = \rho(x_t)V(m)mR^{m-1}. \tag{1}$$

The log-likelihood of the process $N(R, x_t)$ is then given by

$$L(m(x_t), \theta(x_t)) = \int_0^R \log \lambda(r, x_t) dN(r, x_t) - \int_0^R \lambda(r, x_t) dr,$$

where $\theta(x_t) := \log \rho(x_t)$ is the density parameter and the first integral is a Riemann-Stieltjes integral [28]. The maximum likelihood estimators lead to a computation for the local dimension at point x_t, $m(x_t)$, depending on all the neighbors within a distance R from x_t [23]. In practice, it is more convenient to compute a fixed amount k of nearest neighbors. Thus, the local estimators at point x_t are

$$m(x_t) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{R_k(x_t)}{R_j(x_t)} \right]^{-1}, \tag{2}$$

$$\theta(x_t) = \log \left((k-1) / \left(V(m(x_t)) R_k(x_t)^{m(x_t)} \right) \right), \tag{3}$$

where $V(m(x_t)) = (2\pi^{m(x_t)/2})/(m(x_t)\Gamma(\frac{m(x_t)}{2}))$, and $\Gamma(\frac{m(x_t)}{2}) = \int_0^\infty t^{m(x_t)/2-1}e^{-t}dt$. If the data points belong to the same manifold, the authors propose to average over all local estimators $m(x_t)$ in order to obtain a more robust estimator. However, if there are two or more manifolds with different dimensions, the average does not make sense, unless we first cluster according to dimensionality and then estimate the dimensionality for each cluster. Another possibility is to include this in the process via the simultaneous soft clustering and estimation technique described in Section 4. Before this, let us present the proposed framework to naturally handle noise as part of the model.

3 Translated Poisson model

Usually, point samples are contaminated with noise, thus the point process that we observe is not a simple sampling of a low dimensional manifold but a perturbation of this sample process. This can be modeled with a Translated Poisson Process [29], where an underlying (unobservable) point process is translated to an output (observable) point process. The input and output spaces of the points are not necessarily the same or even of the same dimension (clearly, noise brings points outside of the underlying manifold and into the higher dimensional embedding space). More concretely, an input point at location x in the input space X is randomly translated to a location z in the output space Z, according to a conditional probability density $f(z|x)$, called the *transition density*.

For our purposes, we are going to consider the particular case where each point is translated independently of the others and there are no deletions or insertions in the translation process (these more general cases are also studied in [29]). We have the following critical theorem [29] which says that a translated Poisson process is also a Poisson process:

Theorem (Snyder & Miller [29])**.** *Let $\{N(A)\colon A \subseteq X\}$ be a Poisson process with an integrable intensity function $\{\lambda(x)\colon x \in X\}$. Points of this input point process are translated to the output space Z to form the output point process $\{M(B)\colon B \subseteq Z\}$, where each point is independently translated according to the transition density $f(z|x)$. Then, if there are no insertions and deletions, $\{M(B)\colon B \subseteq Z\}$ is a Poisson process with intensity*

$$\mu(z) = \int_X f(z|x)\lambda(x)dx.$$

Since the intensity of the Poisson process in our model is parametrized by the Euclidean distances of the points (and not by the points themselves, see previous Section), we are going to consider a random translation in the distances. This means that we do not observe the original distances but noisy distances. Let $f(s|r)$ be the transition density which defines the random process which translates a distance r in the input space to a distance s in the observable space. If $\lambda(r, x_t)$, defined in (1), is the local rate of the Poisson process which defines the counting process in the input space, then $\mu(s)$, the intensity of the Poisson process in the output space is given by

$$\mu(s, x_t) = \int_0^{R'} f(s|r)e^{\theta}V(m)mr^{m-1}dr. \qquad (4)$$

R' is different from the radius R considered in the counting process $N(R, x_t)$. We consider $R' > R$ in (4) because, points originally at distance greater than R from x_t can be placed within a distance less than R after the translation process. In practice, the maximum translation is small (just a perturbation because of the noise) and we consider $R' = R + \sigma$ in the particular case of a Gaussian transition density (11). The log-likelihood of the translated Poisson process is

$$L(m(x_t), \theta(x_t)) = \int_0^R \log(\mu(s, x_t))dN(s, x_t) - \int_0^R \lambda(r, x_t)dr.$$

The parameters of the maximum log-likelihood are obtained by solving the system of equations $\partial L/\partial m = 0$ and $\partial L/\partial\theta = 0$. We then obtain the following expression for m when we use the k nearest neighbors (k-NN) instead of the points within distance less to R,

$$m(x_t) = \left[\frac{1}{k-1}\sum_{i=1}^{k-1}\frac{\int_0^{R'} f(R_i(x_t)|r)r^{m-1}\log\frac{R_k(x_t)}{r}dr}{\int_0^{R'} f(R_i(x_t)|r)r^{m-1}dr}\right]^{-1}, \qquad (5)$$

where, by an abuse of notation, we have identified $m = m(x_t)$ in the right hand side. Note that this expression reduces to the Levina and Bickel estimator [23] in the particular case that $f(s|r) = \delta(s-r)$, i.e., there is no translation of the original points. This corresponds to the ideal case with no noise.

Equation (5) is a nonlinear recursive expression in m which is difficult to solve. Thus, we are going to approximate it by an easier to compute closed expression. Since the translation density is modeling the effect of noise, the effective support of $f(s|r)$ is going to be concentrated around s. Then, we can substitute r^{m-1} in (5) by its Taylor expansion around R_i. Let us write (5) in the following way

$$m(x_t) = I^{-1} = \left[\frac{1}{k-1}\sum_{i=1}^{k-1} I_i\right]^{-1}, \qquad (6)$$

and expand r^{m-1} in the integral I_i via its Taylor series

$$I_i := \frac{\int_0^{R'} f(R_i|r)r^{m-1}\log\frac{R_k(x_t)}{r}dr}{\int_0^{R'} f(R_i|r)r^{m-1}dr} = \frac{\int_0^{R'} f(R_i|r)\log\frac{R_k(x_t)}{r}dr + \Delta I_{N_i} + \ldots}{\int_0^{R'} f(R_i|r)dr + \Delta I_{D_i} + \ldots} = \frac{I_{N_i}}{I_{D_i}},$$

where

$$\Delta I_{N_i} := (m-1)R_i^{-1}\int_0^{R'} f(R_i|r)(r-R_i)\log\frac{R_k(x_t)}{r}\,dr, \tag{7}$$

and

$$\Delta I_{D_i} := (m-1)R_i^{-1}\int_0^{R'} f(R_i|r)(r-R_i)dr. \tag{8}$$

These integrals are small since the effective support of $f(R_i|r)$ has the same order than the level of noise (considered not very large), and the quantity $(r-R_i)$ is small in the vicinity of R_i. We can then approximate

$$I_i \approx \frac{\int_0^{R'} f(R_i|r)\log\frac{R_k(x_t)}{r}dr}{\int_0^{R'} f(R_i|r)dr}. \tag{9}$$

Notice that with this approximation of I_i, the estimator (6) still reduces to the noise-free Levina-Bickel estimator (2), that is $I_i = \log\frac{R_k}{R_i}$, when $f(R_i|r) = \delta(R_i - r)$. In the more general case, (9) is the expected value of $\log\frac{R_k}{r}$ according to the transition density $f(R_i|r)$ and thus reducing the effect of noise. Using the approximation (9) in (6) we obtain

$$m(x_t) \approx \left[\frac{1}{k-1}\sum_{i=1}^{k-1}\frac{\int_0^{R'} f(R_i|r)\log\frac{R_k}{r}dr}{\int_0^{R'} f(R_i|r)dr}\right]^{-1}. \tag{10}$$

We explicitly estimate, in the following Section, the error produced in $m(x_t)$ when we use the approximation (10) instead of (5), for the particular important case of a Gaussian transition density,

$$f(s|r) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(s-r)^2}{2\sigma^2}\right). \tag{11}$$

In this particular case that the coordinates are perturbed by Gaussian noise, the error in the Euclidean distance can be approximated by a Gaussian as well (see Appendix A for more details). Thus, the expression for the local dimension estimator becomes

$$m(x_t) \approx \left[\frac{1}{k-1}\sum_{i=1}^{k-1}\frac{\int_0^{R'}\exp\left(-\frac{(R_i-r)^2}{2\sigma^2}\right)\log\frac{R_k}{r}dr}{\int_0^{R'}\exp\left(-\frac{(R_i-r)^2}{2\sigma^2}\right)dr}\right]^{-1}. \tag{12}$$

3.1 Approximation error for a Gaussian translation density

In order to estimate the error of approximating (5) by (10), we compute the integrals (7) and (8), which are the largest order error terms of the numerator and denominator, respectively, in the approximation of $m(x_t)$. For the integral (8), notice that the Gaussian is even with respect to R_i and that $(r-R_i)$ is odd. Then, (8) is zero if the effective support of the Gaussian is within the interval $[0, R']$, that is essentially if $R_i \in [3\sigma, R'-3\sigma]$. If $R_i \in [0, 3\sigma]\cup[R'-3\sigma, R']$, (8) is bounded by $4.5\sigma^2(m-1)/R_i$. We will use this bound for ΔI_{D_i} independently of the value of R_i. Regarding the integral (7), we use the Taylor expansion of $(r-R_i)\log\frac{R_k}{r}$ around R_i,

$$(r-R_i)\log\frac{R_k}{r} = (r-R_i)\log\frac{R_k}{R_i} - \frac{(r-R_i)^2}{R_i} + \dots.$$

Again, we consider the worst case scenario, $R_i \in [0, 3\sigma]\cup[R'-3\sigma, R']$, and we obtain

$$\Delta I_{N_i} \le 4.5\sigma^2\frac{m-1}{R_i}\log\frac{R_k}{R_i}.$$

We use these bounds and error propagation theory to obtain the relative error on I_i,

$$\frac{\Delta I_i}{I_i} = \frac{\Delta I_{N_i}}{I_{N_i}} + \frac{\Delta I_{D_i}}{I_{D_i}} = 4.5\sigma^2\frac{m-1}{R_i}\left(\frac{1}{I_{N_i}}\log\frac{R_k}{R_i} + \frac{1}{I_{D_i}}\right),$$

and the relative error on $m_k(x_t)$,

$$\frac{\Delta m(x_t)}{m(x_t)} = \frac{\Delta I}{I} = \frac{1}{I(k-1)}\sum_i \Delta I_i,$$

which is bounded by

$$\frac{\Delta m(x_t)}{m(x_t)} \le \frac{4.5\sigma^2(m(x_t)-1)}{\min_i\left(R_i\tilde{R}_i^{\,m(x_t)-1}\right)}\left(1 + \frac{m(x_t)}{m(x_t,\sigma=0)}\right), \tag{13}$$

where $m(x_t, \sigma=0)$ is (2), or equivalently, (5) with $\sigma = 0$, and $\tilde{R}_i^{\,m-1} = I_{D_i} = \int_0^{R'} f(R_i|r)r^{m-1}dr$. This provides a bound on the error of the approximation for the important case of Gaussian noise. Similar computations can be performed for other translation density (noise models). In the case of $\sigma = 0$ (no noise), the approximation error $\Delta m(x_t)$ is zero, as expected. If we consider $\tilde{R}_i \approx R_i$, the bound (13) is inversely proportional to the signal to noise ratio and proportional to $(m-1)/R_i^{m-2}$, which is a decreasing function of the dimension m for $R_i > 1$. Note that the estimator $m(x_t)$, defined in (5), is invariant to distance rescalings so we can always ensure $R_i > 1$.

4 Dimensionality and density estimation with simultaneous soft clustering

Having introduced the critical translational Poisson model, we are now ready to introduce the mixture of these models

to address the problem of stratification learning for noisy point cloud data. We start with the basic model, and then introduce a regularization term. We conclude the presentation providing asymptotic results.

4.1 Translation Poisson Mixture Model (TPMM)

In [15], we proposed to study a stratification by extending the Levina and Bickel's technique. Instead of modeling each point and its local ball of radius R as a Poisson process and computing the maximum likelihood (ML) for each ball separately, all the possible balls are considered at the same time in the ML function. The probability density function for the whole point cloud becomes a mixture of Poisson distributions with different parameters (dimension and density) in each class. This allows for the presence of different intrinsic dimensions and densities in the dataset. These are automatically computed while being used for soft clustering. We extend this approach here to the more general case when we have mixtures of translated Poisson processes (thereby handling the noise).

Let us consider J different Poisson distributions in the mixture, each one with a (possibly) different dimension m and density parameter θ. Let us denote by ψ the vector set of parameters, $\psi = \{\psi^j = (\pi^j, \theta^j, m^j); j = 1, \ldots, J\}$, where π^j is the mixture coefficient for class j (the proportion of distribution j in the dataset), θ^j is its density parameter ($\rho^j = e^{\theta^j}$), and m^j is its dimension.

The observable event is, as in the Levina-Bickel approach, the number of points inside the ball $B(R, x_t)$ of radius R centered at point x_t, denoted by $y_t = N(R, x_t)$. The total number of observations is T' and $Y = \{y_t; t = 1, \ldots, T'\}$ is the observation sequence. Often, $T' \equiv T$, all points in the dataset are considered. Let us also denote by $p(\cdot)$ the probability density function and by $P(\cdot)$ the probability. The density function of the Poisson mixture model is given by

$$p(y_t|\psi) = \sum_{j=1}^{J} \pi^j p(y_t|\theta^j, m^j).$$

Since the observations follow a Poisson distribution, and we use the translated Poisson model introduced in the previous section, we have

$$p(y_t|\theta^j, m^j) = e^{\int_0^R \log \mu^j(s)\, dN(s,x_t)} e^{-\int_0^R \lambda^j(r) dr},$$

where $\lambda^j(r) = e^{\theta^j} V(m^j) m^j r^{m^j - 1}$ and $\mu^j(s) = \int_0^{R'} f(s|r) e^{\theta^j} V(m^j) m^j r^{m^j-1} dr$. If Y contains T statistically independent variables (a standard assumption), then the probability density function of the observation sequence is the product of the individual probability densities, $p(y_t|\psi)$, and the log-likelihood is

$$L(Y|\psi) = \log p(Y|\psi) = \sum_{t=1}^{T} \log p(y_t|\psi). \tag{14}$$

Let us consider the hidden-state information, that is, which mixture (or expert) generates each observation. We denote by $Z = \{z_t \in C; t = 1, \ldots, T\}$ the set of hidden variables and by $C = \{C^1, C^2, \ldots C^J\}$ the set of class labels. Then, $z_t = C^j$ means that the j-th mixture generates y_t. Using Z we can write the complete data log-likelihood as

$$\log p(Z, Y|\psi) = \sum_{t=1}^{T} \sum_{j=1}^{J} \delta_t^j \log \left[p(y_t|\psi^j) \pi^j \right], \tag{15}$$

where a set of indicator variables δ_t^j, called membership functions, is used in order to indicate the status of the hidden variables:

$$\delta_t^j \equiv \delta(z_t, C^j) = \begin{cases} 1 & \text{if } z_t = C^j, \\ 0 & \text{otherwise.} \end{cases}$$

The unknown parameters in (15) are: The membership function of an expert (class), δ_t^j, the mixture probabilities, π^j, and the parameters of each expert, m^j and θ^j. Usually, problems involving a mixture of experts are solved by the Expectation Maximization (EM) algorithm [10] [21, Chap. 3]. The EM is based on the following decomposition of the log-likelihood (14):

$$\begin{aligned} L(Y|\psi, H) = &\sum_{t=1}^{T} \sum_{j=1}^{J} h^j(y_t) \log \left[p(y_t|\psi^j) \pi^j \right] \\ &- \sum_{t=1}^{T} \sum_{j=1}^{J} h^j(y_t) \log \left[h^j(y_t) \right], \end{aligned} \tag{16}$$

where $H = \{h^j(y_t) \leq 1; t = 1, \ldots, T, j = 1, \ldots, J\}$ and $h^j(y_t)$ is the probability that observation t belongs to mixture j: $h^j(y_t) = E_Z[\delta_t^j | y_t, \psi] = P(\delta_t^j = 1 | y_t, \psi)$, where $E_Z(\cdot)$ is the expectation with respect to Z. Since the membership functions are indicator variables, the first term in (16) is the expectation of (15) with respect to Z. Also notice that the second term is the entropy of the membership functions.

An interesting interpretation of the EM algorithm is introduced in [17], where the EM is seen as an alternate optimization algorithm of the log-likelihood (16). Then, the E-step is nothing else than the maximization of $L(Y|\psi, H)$ with respect to H with the additional constraint that $\sum_{j=1}^{J} h^j(y_t) = 1$ for each observation $t = 1, \ldots, T$. Thus, the variables $h^j(y_t)$ at step $n+1$ of the optimization algorithm are

$$h_{n+1}^j(y_t) = \frac{p(y_t|m_n^j, \theta_n^j) \pi_n^j}{\sum_{l=1}^{J} p(y_t|m_n^l, \theta_n^l) \pi_n^l}. \tag{17}$$

In the same way, variables ψ are obtained by maximizing $L(Y|\psi, H)$ with respect to ψ with an additional constraint for the mixture probabilities: $\sum_{j=1}^{J} \pi^j{=}1$. This gives equations (21)-(23) for the variables at step $n+1$. In order to compute m_{n+1}^j we have used the same approach as in [23], by means of a k nearest neighbor graph. The TPMM approach just described is summarized in **R-TPMM Algorithm** below, for the particular case of $\alpha = 0$ (no regularization, see below).

4.2 Regularized TPMM

The TPMM algorithm seeks a soft clustering according to dimensionality and density, considering noise in the data, but does not (explicitly) take into account spatial information. Adding regularization is the goal of this section. Regularization further helps to improve the classification in noisy data and points lying close to manifold edges (see results in figures 1 and 2). This regularization is inspired in part by the work in [1] for the neighborhood EM (NEM), where the authors extend the EM algorithm adding spatial constraints. This neighborhood spatial information is introduced as a penalization term in the log-likelihood, following Hathaway's EM interpretation [17]. In our context, we complete (16) with a spatial term $S(H)$,

$$F(\psi, H) = L(Y|\psi, H) + \alpha S(H), \tag{18}$$

where α is a parameter that controls the tradeoff between the spatial term and the likelihood. Its value is also related to the amount of noise in the data.[1] Then, function F is maximized with an alternate optimization technique. Since the new term, S, only depends on H, the optimization procedure results in a EM-type algorithm with a modified membership probability that not only depends on the likelihood but also on the spatial criteria. The NEM algorithm uses (note the similitude with MRFs, see below)

$$S_{NEM}(H) = \sum_{t=1}^{T}\sum_{j=1}^{J} h^j(y_t) \sum_{l\sim t} h^j(y_l),$$

where $l \sim t$ indicates that there is a neighborhood relationship between observations l and t. By maximizing this term, we want, for each observation t, as many neighbors as possible with high probability of belonging to the same class as observation t, thus regularizing the classification. However, we will use a more general expression for $S(H)$ based on a dissimilarity measure, $\mathcal{D}$, between every observation and other observations in the sequence,

$$S(H) = -\sum_{t=1}^{T}\sum_{j=1}^{J} h^j(y_t)\mathcal{D}(t, j, X, H). \tag{19}$$

The expression (19) provides a generic framework for introducing constraints in the soft classification, besides the ones already present in the TPMM model, namely dimensionality and density. One possibility, as in the NEM algorithm, is to introduce spatial regularity. Then, as dissimilarity measure we use $\mathcal{D} = \mathcal{D}_R$ defined as

$$\mathcal{D}_R := \sum_{l\sim t}(1 - h^j(y_l))^2.$$

Different neighborhoods definitions in $\mathcal{D}_R$ result in different kinds of regularization. A natural choice is the manifold neighborhood, for that, we can define as neighbors the k nearest neighbors. However, for specific applications one might be interested in other neighborhoods, e.g., pixel neighborhoods or contiguous frames in video applications (see experiment in Figure 8 and Table 5).

We could also impose spatial intra-class compactness with the definition of a proper dissimilarity function, as in [16].

As noted in [1], the EM algorithm with additional constraints can be seen as finding the Gibbs distribution with energy $-F(\psi, H)$. In the particular case when the additional constraint is neighborhood dependent, $S_{NEM}(H)$ and $S(H)$ with $\mathcal{D}_R$, the Gibbs distribution defines a Markov Random Field.

The maximization of F (Equation (18)), is obtained as in [1], with an alternate optimization technique which results in an EM-type algorithm. Maximizing (18) with respect to H, with $S(H)$ defined in (19) – with the constraints $\sum_{j=1}^{J} h^j(y_t) = 1$ for each observation $t = 1, \ldots, T$, by means of Lagrange multipliers – results in the following expression for the membership probabilities:

$$h^j(y_t) = \frac{p(y_t|m^j, \theta^j)\pi_n^j e^{-\alpha\mathcal{D}(t,j,X,H)}}{\sum_{l=1}^{J} p(y_t|m^l, \theta^l)\pi^l e^{-\alpha\mathcal{D}(t,l,X,H)}}. \tag{20}$$

Since the only term in (18) which depends on ψ is $L(Y|\psi, H)$, the optimal values of $\psi^j = \{(\pi^j, \theta^j, m^j)$ for $j = \{1, \ldots, J\}\}$ do not change with respect to the original TPMM algorithm. The regularized version of the TPMM algorithm is summarized in the **R-TPMM Algorithm** below (Regularized Translated Poisson Mixture Model Algorithm).

The EM suffers from local maxima, this can be alleviated running the algorithm several times with different initializations. In particular, we add to the EM iterations an extra loop where the parameters m^j and θ^j of each class are reinitialized every odd iteration and π^j every even iteration.

[1] The study of the possible connection between the regularization factor α and the level of noise and the translation density in the translation Poisson model is an interesting subject of future research. Note that this regularization is important beyond the noise, e.g., at manifolds edges, see experimental results.

R-TPMM Algorithm

REQUIRE: The point cloud data, J (number of desired classes), k (scale of observation), α (regularization parameter), and σ (noise level or full noise/translation function f).
ENSURE: Regularized soft clustering according to dimensionality and density.

1. Compute the local estimators

$$m(x_t) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \frac{\int_0^{R'} f(R_i(x_t)|r) \log \frac{R_k(x_t)}{r} dr}{\int_0^{R'} f(R_i(x_t)|r) dr} \right]^{-1}$$

$$\theta(x_t) = \log \left((k-1) / \left(V(m(x_t)) R_k(x_t)^{m(x_t)} \right) \right)$$

   In particular, we use the definition of f given in (11).

2. Initialize $\psi_0 = \{\pi_0^j, m_0^j, \theta_0^j\}$ and $\bar{\psi}_0 = \{\bar{\pi}_0^j, \bar{m}_0^j, \bar{\theta}_0^j\}$ to any set of values which ensures that $\sum_j \pi_0^j = \sum_j \bar{\pi}_0^j = 1$ and $\bar{H}_0 = \{\bar{h}_0^j(y_t) = 1/J;\ j = 1, \ldots, J,\ t = 1, \ldots, T\}$.

3. Iterations on l,

   3A. If l odd
   Set $\bar{m}_l^j = m_0^j$ and $\bar{\theta}_l^j = \theta_0^j$, for all $j = 1, \ldots, J$.
   Else
   Set $\bar{\pi}_l^j = 1/J$, for all $j = 1, \ldots, J$.

   3B. Iterations on n,

   For all $j = 1, \ldots, J$:

   3B.1: Compute, for all $t = 1, \ldots, T$,

$$h_{n+1}^j(y_t) = \frac{p(y_t|m_n^j, \theta_n^j) \pi_n^j e^{-\alpha \mathcal{D}(t,j,X,H_n)}}{\sum_{l=1}^J p(y_t|m_n^l, \theta_n^l) \pi_n^l e^{-\alpha \mathcal{D}(t,l,X,H_n)}},$$

   where $H_n = \{h_n^j(y_t);\ j = 1, \ldots, J,\ t = 1, \ldots, T\}$.

   3B.2: Compute

$$\pi_{n+1}^j = \frac{1}{T} \sum_{t=1}^T h_n^j(y_t) \tag{21}$$

$$m_{n+1}^j = \left[\sum_{t=1}^T h_n^j(y_t) m(x_t)^{-1} / \sum_{t=1}^T h_n^j(y_t) \right]^{-1} \tag{22}$$

$$\rho_{n+1}^j = e^{\theta_{n+1}^j} = \left[\sum_{t=1}^T h_n^j(y_t) f(x_t)^{-1} / \sum_{t=1}^T h_n^j(y_t) \right]^{-1} \tag{23}$$

   where $\rho(x_t) = e^{\theta(x_t)}$.

   Until convergence of ψ_n, that is, when $||\psi_{n+1} - \psi_n||_2 < \epsilon$, for a certain small value ϵ.

   Set $\bar{\psi}_{l+1} = \psi_n$ and $\bar{H}_{l+1} = H_n$.

   Until $||\bar{\psi}_{l+1} - \bar{\psi}_l||_2 < \epsilon$, $||\bar{H}_{l+1} - \bar{H}_l||_2 < \epsilon$ or $l = l_{\max}$.[2]

[2] In the experiments we use $l_{\max} = 10$

Remark 1. *The PMM and R-PMM algorithms introduced respectively in [15] and [16] are particular cases of the parameters α (regularization) and σ (noise) in the R-TPMM algorithm . Let us introduce the following notation for the particular cases of these parameters:*

- *PMM:* $\alpha = 0$ *and* $\sigma = 0$.
- *R-PMM:* $\alpha > 0$ *and* $\sigma = 0$.
- *TPMM:*$\alpha = 0$ *and* $\sigma > 0$.
- *R-TPMM:*$\alpha > 0$ *and* $\sigma > 0$.

We will use the above notation in the experiments in Section 5.

Remark 2. *Notice that the estimators* (22)-(23) *in the PMM and R-PMM approaches ($\sigma = 0$) are weighted harmonic means of the local estimators* (2)-(3) *of Levina-Bickel. The weight at each point is the probability of the membership function, h. In the particular case of a unique class, $J = 1$, we obtain the global dimension estimator proposed by MacKay and Ghahramani (http://www.inference.phy.cam.ac.uk/mackay/dimension/), a particular case of our proposed framework.*

As proved in [2], if α is small enough, (18) has a guaranteed global maximum for a fixed value of ψ, and the additional term $S(H)$ does not affect the convergence of the EM-type algorithm. It can be shown (see Appendix B) that, for the case of $\mathcal{D}_R$, the corresponding bound on α is

$$\alpha_R < \frac{1}{2 \max_{t,j} \sum_{s \sim t} (1 - h^j(x_s))}.$$

Notice that $\alpha_R < 1/(2k)$ in the worst case scenario.

Using the same analysis as in Section 3.1 we find that the relative error produced in (22) by using the approximation (10) for $m(x_t)$ is

$$\frac{\Delta m^j}{m^j} \leq \frac{4.5\sigma^2(m^j - 1)}{\min_{i,t} \left(R_i(y_t) \tilde{R}_i(y_t)^{m^j - 1} \right)} \left(1 + \frac{m^j}{m^j(\sigma = 0)} \right),$$

where $m^j(\sigma = 0)$ is (22) with $\sigma = 0$, and $\tilde{R}_i(y_t)^{m^j - 1} = I_{D_i}(y_t)$.

4.3 Asymptotic analysis

Levina and Bickel show in [23] that under the assumptions $T \to \infty$, $k \to \infty$, and $k/T \to 0$, that is when the Poisson approximation is correct, the mean and variance of the dimension estimator (2) (with $k - 2$ instead of $k - 1$ in the denominator) are

$$E[m(x_t)] = \mathrm{m}_T, \quad \mathrm{Var}[m(x_t)] = \frac{\mathrm{m}_T^2}{k - 3},$$

where m_T is the actual dimension. We can apply the same type of analysis to our model in the particular case of hard clustering, that is

$$h^j(y_t) = \begin{cases} 1 & \text{if } j = \text{argmax}_i h^i(y_t), \\ 0 & \text{otherwise.} \end{cases}$$

We assume, in addition, that all the points that belong to class j are well classified. Then, we obtain the following results

$$E[m^j] = \mathrm{m}_T^j + \frac{\mathrm{m}_T^j}{(k-1)N_j - 1},$$

$$\mathrm{Var}[m^j] = (\mathrm{m}_T^j)^2 O\left(\frac{1}{(k-1)N_j - 4}\right),$$

where m_T^j is the correct intrinsic dimension of class j and N_j is the amount of points classified as class j. See Appendix C for the details of the proof. The analysis of the density estimator θ^j is the subject of current research, as it is the study of the asymptotic behavior for the full soft clustering model.

5 Experimental results

We now present experimental results with synthetic and real data for the proposed R-TPMM and its variants. We also compare some of the results with the ones obtained with GPCA [33] and the Souvenir and Pless [30] algorithms. We fixed α and σ experimentally. For α we usually use values in the interval $[0, 3]$ except for the video experiment with temporal regularization where we use $\alpha = 40$. As for the case of σ we use a value in the order of the mean distance to the first neighbor: $\sigma = \nu \bar{R}_1$, where $\bar{R}_1 = \frac{1}{N}\sum_t R_1(x_t)$ and $0 \leq \nu \leq 1$. In the experiments with real data – digits, faces, video activities, and motion – we use the following values for ν: 0.4,0.4, 0.25, and 1 respectively. In the first (artificial data) experiment, since we know the level of noise in the point coordinates, we use the estimated σ as computed in Appendix A. The only parameter in GPCA is the number of clusters. In the Souvenir-Pless algorithm the input parameters are the number of nearest neighbors and the dimension of each cluster. We also fixed these parameters experimentally in order to obtain the best classification results.

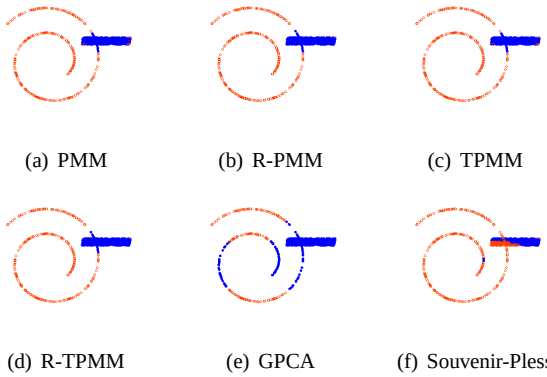
First, we work with a synthetic point cloud data formed by 300 samples of a spiral and 800 of a plane, both in 3D embedding space. We compare the following algorithms: PMM, R-PMM, TPMM, R-TPMM, GPCA, and Souvenir-Pless. Figure 1 shows, for each algorithm, the point cloud with each point colored and marked differently according to its classification. In the different versions of our proposed algorithm we set $k = 30$, $J = 2$, $\alpha = 0.5$, and $\sigma = 0.1$.

	PMM		R-PMM		TPMM		R-TPMM	
	Estimated parameters							
m	1.90	1.02	1.90	1.00	1.87	1.03	1.87	1.01
θ	1.01	1.10	1.00	1.13	1.05	1.09	1.03	1.11
	Points in each class							
Pl.	787	13	798	2	788	12	798	2
Sp.	21	279	22	278	21	279	22	278

Table 1: *Estimated parameters and clustering results of a spiral and a plane ($k = 30$, $J = 2$).*

We test TPMM and R-TPMM with a small value of σ different than zero even if there is no noise just to show that a small error in the estimation of σ does not significantly affect the result. Notice that the regularized versions of our proposed algorithm improve the classification at the edges. In the Souvenir-Pless algorithm we use $k = 10$ and dimensions 2 and 2 (it gives a better result than using the actual dimensions, 2 and 1, as parameters). The GPCA algorithm does not give good results because it is designed for linear manifolds. Table 1 contains quantitative results of the different versions of our algorithm.

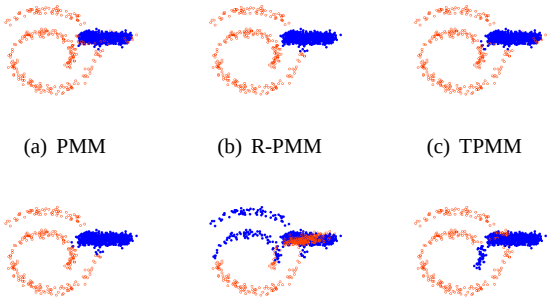
Next, we added Gaussian noise, with standard deviation 0.66, to the point coordinates. Then, if we approximate the transition density with a Gaussian (see Appendix A), we use the estimated standard deviation $\sigma = 0.66\sqrt{2} = 0.93$. The rest of the parameters we use are $k = 40$, $J = 2$, $\alpha = 2$, and for Souvenir-Pless, $k = 20$ and dimensions 2 and 2. The qualitative comparison of the different algorithms can be seen in Figure 2. Again, notice how the classification of the points at the edges is better in the regularized versions. Table 2 contains the quantitative results for the different variants of the proposed algorithm. In particular, it can be seen that the translated versions give an estimation for the dimension m less sensible to noise.



(a) PMM (b) R-PMM (c) TPMM

(d) R-TPMM (e) GPCA (f) Souvenir-Pless

Figure 1: *Clustering of a spiral and a plane. Results with different algorithms (this is a color figure).*



(a) PMM (b) R-PMM (c) TPMM

(d) R-TPMM (e) GPCA (f) Souvenir-Pless

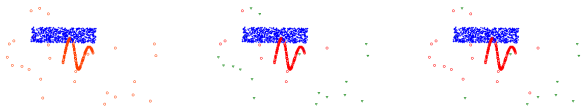
Figure 2: *Clustering of a spiral and a plane with noise. Results with different algorithms (this is a color figure).*

	PMM		R-PMM		TPMM		R-TPMM	
	Estimated parameters							
m	2.47	1.51	2.48	1.43	1.86	1.35	1.87	1.32
θ	0.13	0.03	0.15	0.03	0.87	0.34	0.83	0.40
	Points in each class							
Pl.	764	36	800	0	784	16	800	0
Sp.	22	278	25	275	27	273	29	271

Table 2: *Estimated parameters and clustering results of a spiral and a plane with noise ($k = 40$, $J = 2$).*

In order to see how the R-TPMM performs in the presence of outliers, we have perturbed 2.5% of the points in the spiral and plane. The original point coordinates are within the intervals $[-11, 21]$, $[5, 25]$, and $[-11, 14]$. The perturbed points follow a uniform distribution within the intervals $[-30, 30]$, $[-15, 35]$, and $[-30, 30]$. We use $\alpha = 1$ and $\sigma = 0.1$. Figure 3 shows the classification results for three different cases: a) $J = 2$, $k = 30$; b) $J = 3$, $k = 20$; b) $J = 3$, $k = 30$. The dimensions obtained in these three cases are: a) 1.10 and 1.88; b) 1.06, 1.88, and 14.30; c) 1.05, 1.85, and 11.16. When we set two classes, the outliers are classified as the same class as the spiral. Note that the estimation of the embedding dimensions are not affected by the outliers. When we set three classes the class 'outlier' has a larger dimension and the amount of outliers which belong to this class is reduced when k increases. Since we do not have enough samples of the class 'outlier' in each ball (there are mixed samples from the spiral and/or the plane), we obtain a very large dimension. In these balls, the assumption of approximate constant density is not satisfied either.

The experiment in Figure 4 illustrates how the soft clustering is done according to both dimensionality and density. The data consists of 2000 points on the Swiss roll, 400 on a line with high density and 50 on another less dense line. We have set $J = 3$ and the algorithm clusters the line in



(a) $J = 2, k = 30$ (b) $J = 3, k = 20$ (c) $J = 3, k = 30$

Figure 3: *Clustering of a spiral and a plane with 2.5% of outliers with R-TPMM, $\alpha = 1$ and $\sigma = 0.1$ (this is a color figure).*

two different classes, according to the different densities. The estimated dimensions are: 1.98, 1.02 and 0.99. And the estimated densities: 0.49, 0.53 and 6.89 respectively.

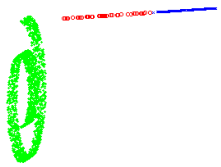
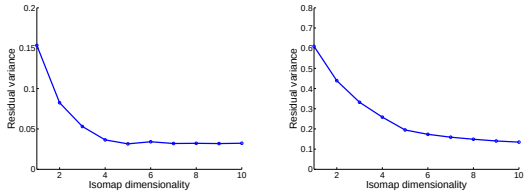


Figure 4: *Clustering of a Swiss roll and a line with two different densities with R-TPMM, $k = 20$, $J = 3$, and $\alpha = 2$ (this is a color figure).*

As a test of the performance with real data, we first work with the MNIST database of handwritten digits,[3] which has a test set of 10.000 examples. Each digit is an image of 28×28 pixels and we treat the data as 784-dimensional vectors. We analyze the mixture of digits one and two, some examples of those scanned digits as well as the clustering results are in Figure 5. Observe how the classification improves adding regularization and including the noise in the model (Translated Poisson). We have used R-PMM with $\alpha = 3$, TPMM with $\sigma = 1.5$, and R-TPMM with $\alpha = 2$ and $\sigma = 1.5$. Levina-Bickel's technique gives a dimension value of 11.26 and Costa-Hero's 9. These methods give a dimension in between the two different dimensions present in the point cloud. With the R-TPMM algorithm (and its variants), we are able to separate the points (images) corresponding to each digit, both sets have different dimensionality and density, and handle the noise and regularization. We have observed that some other digits do have the same dimensionality, as expected. Observe in the Table of Fig. 5 how the dimension is reduced with the (R-)TPMM, these values are much closer (than the ones with (R-)PMM) to the dimension obtained with Isomap, see graph in Figure 6, applied to each one of the digits by separate. The fact that

[3] http://yann.lecun.com/exdb/mnist/

the dimension is reduced when considering the translated process indicates that the high dimensions were originally due to the noise (this can be also inferred by observing the Isomap eigenvalues in Fig.6).



(a) Digit '1' (b) Digit '2'

Figure 6: *Isomap dimensionality of Digits one and two. The graph show the residual variance of the first ten Isomap embedding dimensionalities.*

We also analyze images from the Yale Face Database B,[4] which contains images of 10 subjects under 585 viewing conditions (9 poses and 65 illumination conditions), see Fig. 7. Each image has a size of 640×480 pixels. For computational reasons we subsampled the images by a factor of ten and use each 64×48 image as a vector in a high dimensional space. We analyze the point cloud formed by the 585 images of subject 5 (varying pose and illumination) together with the 65 images of subject 6 only in the first pose and under varying illuminations. The estimated dimensions and confusion matrices using the PMM and R-TPMM algorithm with $\alpha = 0.25$ and $\sigma = 1$ are presented in Table 3. Note how both subjects are well separated, and the set of images of subject 5 has a dimension one unity larger than the dimension for subject 6, since we do not consider the pose variation for this subject. The classification results are improved using regularization and the translated Poisson model. Observe also that changing the number of k nearest neighbors does not significantly change the results. Table 4 contains the confusion matrix obtained with the GPCA and the Souvenir-Pless algorithms. These algorithms are computed with a pre-projection of the data onto a 5-dimensional space[5]. This is necessary in the GPCA because, although not being an iterative algorithm, it consumes a lot of time in high dimensional spaces. For the Souvenir-Pless algorithm this point is not so critical but we obtained better classification results in the reduced dimensionality space. However, with the proposed R-TPMM we obtain better results in the original space.

It must be clarified that the R-TPMM is able to separate both subjects because their corresponding images lie in manifolds of different dimensions. However, if we consider,

[4]http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

[5]We compute the SVD of the matrix data $I = U\Sigma V^T$ and consider the matrix formed by the first 5 columns of V^T as the embedded data.

	PMM				R-TPMM			
	$k = 35$		$k = 50$		$k = 35$		$k = 50$	
	Estimated dimension							
m	4.10	2.94	4.37	2.79	3.34	2.59	3.60	2.59
	Points in each class							
S. 5	569	16	575	10	584	1	584	1
S. 6	0	65	0	65	0	65	0	65

Table 3: *Clustering results of the mixture of subject 5 (all poses, all illuminations) and subject 6 (one pose, all illuminations) in the Yale Face Database B. PMM and R-TPMM ($\alpha = 0.25$, $\sigma = 1$) algorithms with two different values of k. The algorithms are applied in the 64×48 dimensional space.*

	GPCA		Souvenir-Pless	
	Points in each class			
Subject 5	325	260	476	109
Subject 6	0	65	20	45

Table 4: *Clustering results of the mixture of subject 5 (all poses, all illuminations) and subject 6 (one pose, all illuminations) in the Yale Face Database B. We apply GPCA and Souvenir-Pless algorithms to the data pre-projected onto a 5 dimensional space.*

for example, a fixed pose under varying illuminations, in both subjects, all the points are classified in the same class since both manifolds have the same dimension (complexity). In this particular case, we tested the GPCA algorithm and it gives a 100% accurate classification.

Figure 7: *Examples of images of subjects 5 and 6 of the Yale Face Database B. See results in Table 3.*

The R-TPMM framework is also tested to study different human activities in video. We created a point cloud with the frames of a video of a person performing four different activities: Standing, walking, jumping, and arms waving, all performed in a static background. Each original frame is 480×640, sub-sampled to 48×64 pixels, with 1673 frames (see some frame examples in Figure 8). This is mainly to speed up computations. In video applications, one may be interested in temporal regularization. For that, we consider a temporal neighborhood in $\mathcal{D}_R$, more concretely we take into account the 6 previous and 6 posterior frames in the

	PMM		R-PMM		TPMM		R-TPMM	
	Estimated parameters							
m	7.33	12.79	7.34	12.87	2.86	7.14	2.88	7.24
θ	-7.38	-23.99	-7.50	-23.11	-1.52	-12.70	-1.62	-12.90
	Points in each class							
'1'	1032	0	1032	0	1032	0	1029	3
'2'	70	1065	57	1078	36	1099	17	1118

Figure 5: *Clustering of scanned digits '1' and '2.' Some examples of digits and table with estimated parameters and clustering results for different variants of the R-TPMM algorithm with J=2, k=30 (recall that the density is $\rho = e^{\theta}$ and thus $\rho \geq 0$ for $\theta \in \mathbb{R}$).*

regularization term. The confusion matrix with the classification results using the R-TPMM algorithm (with $k = 10$, $\alpha = 40$ and $\sigma = 0.25$) is presented in Table 5. The error in the classification affects only 4% of the frames.

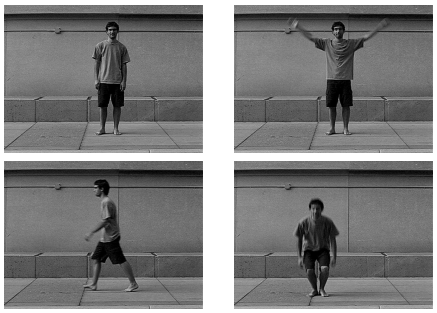


Figure 8: *Four sample frames of human activities in video.*

Samples in each cluster				
	C1	C2	C3	C4
Standing	505	0	6	0
Walking	0	464	45	14
Waving	1	1	430	0
Jumping	0	0	0	207

Table 5: *Classifying human activities in video with the R-TPMM algorithm ($k = 10$, $\alpha = 40$ and $\sigma = 0.25$). We use the 6 previous and 6 posterior frames as neighbors in $\mathcal{D}_R$, which results in a temporal regularization. The global classification is 96% accurate.*

Finally, we tested the R-TPMM algorithm in a motion segmentation application. We use a sequence of the Kanatani Lab, [6] see some examples of frames in Figure 9. This sequence was originally used in [19] and then in [34]. The data consists of the 2D projection coordinates of the trajectories along the sequence of some interest points. The sequence that we analyze corresponds to a car moving in a parking lot and there are two different motions in the sequence. As in [34] we pre-project the data, originally in a 60-dimensional space (2 coordinates $\times$ 30 frames), onto a 5-dimensional space. In Table 6 we show the classification effectiveness for different methods: Costeira-Kanade, Ichimura, Kanatani-Sugaya (the three of them reported in [19]), Souvenir-Pless, GPCA and R-TPMM. For the R-TPMM we use $k = 10$, $\alpha = 2$ and $\sigma = 0.05$. We also tested our algorithm with the other two sequences used in [19, 34] and obtained a single class since the two different motions have the same dimension (complexity). Thus, it is necessary to introduce an additional constraint in the R-TPMM approach in order to deal with these cases.

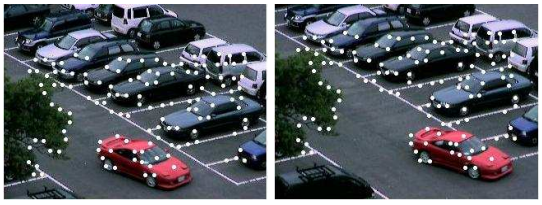


Figure 9: *Two frames of a sequence of the motion segmentation database of the Kanatani Laboratory.*

Method	Effectiveness
Costeira-Kanade	60.3%
Ichimura	92.6%
Kanatani-Sugaya	100%
Souvenir-Pless	93.38%
GPCA	100%
RTPMM	100%

Table 6: *Classification rates, using different methods, for the motion segmentation in the Kanatani Laboratory sequence (see example frames in Fig. 9).*

Regarding the computational time, the most expensive part is the kNN-graph. In the digits experiment (Fig. 5), 2167 points of dimension 784, the execution takes 18.37s while 10.29s of the total time is spent in the computation of the kNN-graph. For the experiment with the Yale faces (Fig.

[6] http://www.suri.it.okayama-u.ac.jp/data.html

7, 650 points of dimension 3072) the execution time is 7.64s (3.70s for the kNN-graph). In the video experiment (Fig. 8, 1673 points of dimension 3072) the total time and the kNN-graph time are, respectively, 29.78s and 24.87 (CPU: Pentium Core 2 Duo, 2.0 GHz, 2.0 GB memory).

6 Conclusions

In this paper we developed a framework for the simultaneous and regularized/constrained estimation of the intrinsic dimensionality and density of high dimensional noisy point cloud data sampled from a stratification, as the basis for complexity/density based soft-clustering. The algorithm is based on a statistical model which addresses the presence of noise in the measurements. Our previous related works [15, 16] are particular cases of the R-TPMM algorithm introduced in this paper. We showed that regularization constraints can be naturally introduced in this approach. The experiments showed the importance of incorporating the noise in the model and also of adding regularization in the classification. We also showed that the algorithm is robust to outliers. With the proper dissimilarity function and neighborhood type, we are able to add spatial or temporal regularity in the classification or intra-class spatial compactness. Other type of constraints are possible under the same proposed framework. Asymptotic theoretical results were also presented.

We would like to follow this direction of work and study other constraints which can be useful for stratification learning. One possibility is to define a dissimilarity function which leads to separate different manifolds that share the same dimensionality and density. This will define a new constraint that will also help in the classification process when there is an intersection of two manifolds (and where the algorithm fails at the present stage). Since the density depends on the dimension, we are intrinsically giving more importance to the dimension criterion in our framework. The control of the relative importance of these two criteria needs also to be addressed.

Appendix A: Estimation of the distribution of distance errors

In this section we derive the distribution of the error in the distance between a pair of points when this distance is computed from noisy points. We are interested in the particular case when the noise follows an i.i.d. Gaussian distribution in each of the point coordinates.

Let $X = \{x_t \in \mathbb{R}^p; t = 1, \ldots, T\}$ and $\hat{X} = \{\hat{x}_t \in \mathbb{R}^p; t = 1, \ldots, T\}$ be two point clouds which are related in the following way: $\hat{x}_t = x_t + n_t$, for each index t, where $n_t \sim N(0, \sigma^2)$, i.e., $\hat{X}$ is a noisy version of X. Let D_{ij} (resp. $\hat{D}_{ij}$) be the Euclidean distance between points x_i and x_j (resp. $\hat{x}_i$ and $\hat{x}_j$). We can write $\hat{D}_{ij}$ as a function of the original points x_i and x_j:

$$\begin{aligned}\hat{D}_{ij} =& ||\hat{x}_i - \hat{x}_j||_2 \\ =& \left(D_{ij}^2 + ||n_i - n_j||_2^2 + 2 <(x_i - x_j), (n_i - n_j)>\right)^{1/2}.\end{aligned}$$

Expanding the previous expression in a Taylor series around D_{ij} (considering the rest of the terms sufficiently small), we obtain,

$$\begin{aligned}\hat{D}_{ij} \approx& D_{ij} + \frac{<(x_i - x_j), (n_i - n_j)>}{D_{ij}} + \frac{||n_i - n_j||_2^2}{2D_{ij}} \\ & - \frac{1}{8}\frac{\left(<(x_i - x_j), (n_i - n_j)>\right)^2}{D_{ij}^3} + O(\sigma^3) \\ =& D_{ij} + D_{n_1} + D_{n_2} + D_{n_3} + O(\sigma^3).\end{aligned}$$

In order to estimate the probability density function of the three error terms D_{n_i}, $i = 1 \ldots 3$, in $\hat{D}_{ij}$ we make use of the following properties:

1. If $X \sim N(\mu, \sigma^2)$ and $a, b \in \mathbb{R}$, then $aX+b \sim N(a\mu+b, (a\sigma)^2)$.
2. If $X \sim N(\mu_X, \sigma_X^2)$ and $Y \sim N(\mu_Y, \sigma_Y^2)$ are independent variables, then:
   (a) $X + Y \sim N(\mu_X + \mu_Y, \sigma_X^2 + \sigma_Y^2)$,
   (b) $X - Y \sim N(\mu_X - \mu_Y, \sigma_X^2 + \sigma_Y^2)$.
3. If $X_1, \ldots, X_p$ are iid variables s.t. $X_i \sim N(\mu_i, \sigma_i^2)$, then $U = \sum_{i=1}^p \left(\frac{X_i - \mu_i}{\sigma_i}\right)^2$ follows a Chi-square distribution with p degrees of freedom, $U \sim \chi_p^2$.
4. If X is a random variable with probability density function $f(x)$ and $Y = aX$, where $a \in \mathbb{R}$, then, the probability density function of Y is $\frac{1}{|a|}f(\frac{x}{a})$.
5. The probability density function of the sum of two independent random variables X and Y with probability density functions f and g is the convolution
$$(f * g)(x) = \int_{-\infty}^{\infty} f(y)g(x - y)\, dy.$$

The error term $D_{n_1} \sim N(0, 2\sigma^2)$, by using properties 1, $2(a)$ and $2(b)$ (notice that the denominator cancels out the weights in the numerator when adding the individual (constant) variances in each coordinate). The second term, $D_{n_2} \sim \hat{\chi}_p^2 = \frac{D_{ij}}{\sigma^2}\chi_p^2\left(\frac{D_{ij}}{\sigma^2}x\right)$ (properties 2(b) and 3). And for the last term, using properties 1 - 4, $D_{n_3} \sim \breve{\chi}_1^2 = \frac{32D_{ij}^2}{\sigma^2}\chi_1^2\left(-\frac{32D_{ij}^2}{\sigma^2}x\right)$.

Finally, using the previous results and Property 5, we can write

$$\hat{D}_{ij} \approx D_{ij} + W; \quad \text{where } W \sim N(0, 2\sigma^2) * \hat{\chi}^2_p * \check{\chi}^2_1.$$

In Figure 10 we show the distribution $N(0, 2\sigma^2)$ compared with the estimated distribution W for $\sigma = 0.5$, $p = 3$ and two different values for D_{ij}: 1.0 and 3.0. As we can see in this Figure, for a fixed σ, as D_{ij} gets larger, the distribution W is closer to a $N(0, 2\sigma^2)$ distribution. Then, for values $\frac{D_{ij}}{\sigma}$ not very small, that is, for sufficient SNR, we can approximate the probability density function of the error in the distance as a Gaussian.

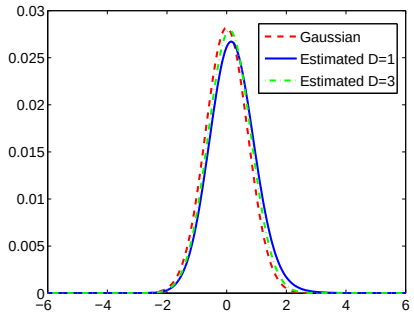


Figure 10: Gaussian distribution, $N(0, 2\sigma^2)$ with $\sigma = 0.5$, compared to the estimated distribution W for two different values of D_{ij}: 1.0 and 3.0.

Appendix B: Bound on α for convergence

We now show that, for a fixed ψ, $F(\psi, H)$ defined in (18) has a global maximum. For that, we follow the same lines as in [2]. Let us call $F_\psi(H)$ the functional (18) when ψ is fixed. $F_\psi(H)$ has a global maximum if it is strictly concave, i.e. if its Hessian matrix $\mathcal{H}$, with components

$$\mathcal{H}_{il,jt} = \frac{\delta^2 F}{\delta h^i_l \delta h^j_t} = \begin{cases} -1/h^j_t & \text{if } i = j \text{ and } l = t \\ 2\alpha(1 - h^j_l) & \text{if } i = j \text{ and } l \sim t \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

is strictly negative. The Gerschgorin-Hadamard Theorem tell us that the eigenvalues λ of this Hessian matrix belong to the union of discs indexed by (j, t) and defined by

$$|\lambda - \mathcal{H}_{jt,jt}| \leq \sum_{(i,l)\neq(j,t)} |\mathcal{H}_{jt,il}|.$$

Substituting the last expression with values in (24) gives

$$\left|\lambda - \frac{1}{h^j_t}\right| \leq \sum_{l\sim t} 2\alpha(1 - h^j_l) \leq 2\alpha \max_t \sum_{l\sim t}(1 - h^j_l).$$

Since $h^j_t \in [0, 1]$, $\mathcal{H}$ is strictly negative, i.e., every eigenvalue $\lambda < 0$, if $\left|\lambda - \frac{1}{h^j_t}\right| < 1$, and this is true for

$$\alpha < \frac{1}{2 \max_t \sum_{l\sim t}(1 - h^j_l)}. \tag{25}$$

In the particular case of hard clustering (25) becomes

$$\alpha < \frac{1}{2 \max_t(\#\text{ neighbors of } t \text{ in other class})},$$

and in the worst case, $\alpha < 1/(2k)$.

Appendix C: Proof of the asymptotic analysis

When we consider the particular case of hard clustering we have

$$h^j(y_t) = \delta^j_t = \begin{cases} 1 & \text{if } j = \text{argmax}_i h^i(y_t), \\ 0 & \text{otherwise.} \end{cases}$$

The estimator of the dimension in class j can be expressed as

$$m^j = \left[\frac{1}{N_j}\sum_{t=1}^{T} \delta^j_t \frac{1}{k-1}\sum_{i=1}^{k-1} \log \frac{R_k(y_t)}{\bar{R}_i(y_t)}\right]^{-1}, \tag{26}$$

where N_j is the number of points clustered in class j and

$$\log \bar{R}_i(y_t) = \frac{\int_0^{R'} f(R_i(y_t)|r) \log r dr}{\int_0^{R'} f(R_i(y_t)|r) dr}. \tag{27}$$

In the (R-)PMM approach we have $\bar{R}_i = R_i$. We can rewrite (26) as

$$m^j = N_j(k-1)\mathrm{m}^j_T Z^{-1}, \tag{28}$$

where m^j_T is the actual dimension of class j and Z is

$$Z = \sum_{t=1}^{T} \delta^j_t Y_t\,; \quad Y_t = \mathrm{m}^j_T \sum_{i=1}^{k-1} \log \frac{R_k(y_t)}{\bar{R}_i(y_t)}.$$

With the proper definition of the upper limit R' in the integral in (27) and the transition density $f(R_i|r)$ when R_i is close to R', we can guarantee that $\bar{R}_i \leq R_k$ (always true in (R-)PMM). In this case, we use the fact that $(\bar{R}_i/R_k)^{\mathrm{m}^j_T}$

is distributed, under the Poisson assumption, as a Uniform(0,1) distribution, the $-\log$ of such a distribution is an Exponential(1), and then, the sum of $(k-1)$ Exponential(1) distributed variables is a Gamma($k-1$,1). Then, $Y_t \sim$ Gamma($k-1$,1) and the sum of N_j Gamma($k-1$,1) distributions gives $Z \sim$ Gamma($N_j(k-1)$,1) and $Z^{-1} \sim$ Inverse-Gamma($(k-1)N_j$,1). The expectation of Z^{-1} is $1/(N_j(k-1)-1)$, and substituting in (28), considering that $1 < N_j(k-1)$, yields

$$E[m^j] = \mathrm{m}_T^j + \frac{\mathrm{m}_T^j}{(k-1)N_j - 1}.$$

Regarding the variance,

$$\mathrm{Var}[m^j] = N_j^2(k-1)^2\mathrm{Var}[Z^{-1}],$$

where

$$\mathrm{Var}[Z^{-1}] = \frac{1}{(N_j(k-1)-1)^2(N_j(k-1)-2)}.$$

We now define

$$a := \frac{2 - 5N_j(k-1)}{N_j^2(k-1)^2(N_j(k-1)-2)}.$$

After simple computations and under the hypothesis that $|a| < 1$, we obtain

$$\mathrm{Var}[m^j] = \frac{(\mathrm{m}_T^j)^2}{N_j(k-1)-2}\left[1 + \sum_{n=1}^{\infty} a^n\right],$$

and since the second term is smaller than the first one, we can write

$$\mathrm{Var}[m^j] = (\mathrm{m}_T^j)^2 O\left(\frac{1}{N_j(k-1)-2}\right).$$

Acknowledgments This work has been supported by ONR, DARPA, NSF, NGA, ARO, the McKnight Foundation, and the Juan de la Cierva Programme. GH was a postdoctoral associate at Institute of Mathematics and its Applications, University of Minnesota, USA, while performing part of this work. We like to thank the Yale Face Database Project, the Kanatani Lab and the people involved in the GPCA website[7] for making their data and codes publically available. We also thank Richard Souvenir for providing us some data and his code.[8]

References

[1] C. Ambroise and G. Govaert. Clustering of spatial data by the EM algorithm. In *geoENV I - Geostatistics for Environmental Applications*, 1996.

[2] C. Ambroise and G. Govaert. Convergence of an EM-type algorithm for spatial clustering. *Pattern Recognition Letters*, 19(10):919–927, 1998.

[3] D. Barbara and P. Chen. Using the fractal dimension to cluster datasets. In *Proceedings of the Sixth ACM SIGKDD*, pages 260–264, 2000.

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in NIPS 14, Vancouver, Canada*, 2002.

[5] P. Bendich, D. Cohen-Steiner, J. Harer, H. Edelsbrunner, and D. Morozov. Inferring local homology from sampled stratified spaces. In *To appear in 48th Annual IEEE Symposium on Foundations of Computer Science*, 2007.

[6] M. Brand. Charting a manifold. In *Advances in NIPS 16, Vancouver, Canada*, 2002.

[7] W. Cao and R. Haralick. Nonlinear manifold clustering by dimensionality. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 1, pages 920–924, 2006.

[8] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis: Special issue on Diffusion Maps and Wavelets*, 21:5–30, 226.

[9] J. A. Costa and A. O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Trans. on Signal Processing*, 52(8):2210–2221, 2004.

[10] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data. *Journal of the Royal Statistical Society Ser. B*, 39:1–38, 1977.

[11] T. K. Dey, J. Giesen, S. Goswami, and W. Zhao. Shape dimension and approximation from samples. *Discrete and Comput. Geom.*, 29:419–434, 2003.

[12] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533, 2002.

[13] A. Gionis, A. Hinneburg, S. Papadimitriu, and P. Tsparas. Dimension induced clustering. In *Proceeding of the Eleventh ACM SIGKDD*, pages 51–60, 2005.

[14] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *Proceedings of CVPR*, 2007.

[15] G. Haro, G. Randall, and G. Sapiro. Stratification learning: Detecting mixed density and dimensionality in high dimensional point clouds. In *Advances in NIPS 19, Vancouver, Canada*, 2006.

[16] G. Haro, G. Randall, and G. Sapiro. Regularized mixed dimensionality and density learning in computer vision. In *Proceedings of 1st Workshop on Component Analysis Methods for Classification, Clustering, Modeling and Estimation Problems in Computer Vision, in conjunction with CVPR*, Minneapolis, June 2007.

[17] R. Hathaway. Another interpretation of the EM algorithm for mixture distributions. *Statistics and Probability Letters*, 4(2):53–56, 1986.

[18] K. Huang, Y. Ma, and R. Vidal. Minimum effective dimension for mixtures of subspaces: A robust GPCA algorithm and its applications. In *Proceedings of CVPR*, pages 631–638, 2004.

[7] http://perception.csl.uiuc.edu/gpca/

[8] http://www.cs.wustl.edu/∼rms2/kmanifolds.htm

[19] K. Kanatani and Y. Sugaya. Multi-stage optimization for multi-body motion segmentation. In *Proceedings of the Australia-Japan Advanced Workshop on Computer Vision*, pages 25–31, September 2003.

[20] B. Kegl. Intrinsic dimension estimation using packing numbers. In *Advances in NIPS 14, Vancouver, Canada*, 2002.

[21] S. Y. Kung, M. W. Mak, and S. H. Lin. *Biometric Authentication: A Machine Learning Approach*. Prentice Hall, 2004.

[22] S. Lafon, Y. Keller, and R. R. Coifman. Data fusion and multi-cue data matching by diffusion maps. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 28(11):1784–1797, 2006.

[23] E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in NIPS 17, Vancouver, Canada*, 2005.

[24] L. Lu and R. Vidal. Combined central and subspace clustering for computer vision applications. In *Proceedings of the 23rd International Conference on Machine Learning*, volume 148, pages 593–600, 2006.

[25] P. Mordohai and G. Medioni. Unsupervised dimensionality estimation and manifold learning in high-dimensional spaces by tensor voting. In *IJCAI*, page 798, 2005.

[26] M. Polito and P. Perona. Grouping and dimensionality reduction by locally linear embedding. In *Advances in NIPS 14, Vancouver, Canada*, 2002.

[27] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[28] D. L. Snyder. *Random Point Processes*. Wiley, New York, 1975.

[29] D. L. Snyder and M. I. Miller. *Random Point Processes in Time and Space*. Springer-Verlag, 1991.

[30] R. Souvenir and R. Pless. Manifold clustering. In *ICCV*, pages 648–653, 2005.

[31] F. Takens. On the numerical determination of the dimension of an attractor. *Lecture notes in mathematics. Dynamical systems and bifurcations*, 1125:99–106, 1985.

[32] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[33] R. Vidal, Y. Ma, and J. Piazzi. Generalized principal component analysis (GPCA). In *Proceedings of CVPR*, pages 621–628, 2003.

[34] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(12), 2004.