An Ellipsoidal K-Means for Document Clustering

Fabon Dzogang, Christophe Marsala, Marie-Jeanne Lesot, Maria Rifqi

An ellipsoidal K-means for document clustering

Fabon Dzogang*, Christophe Marsala*, Marie-Jeanne Lesot* and Maria Rifqi†
*LIP6, Université Pierre et Marie Curie-Paris 6, UMR7606, Paris, France
†LIP6 & Université Panthéon-Assas, Paris, France
Contact email: fabon.dzogang@lip6.fr

***Abstract*—We propose an extension of the spherical K-means algorithm to deal with settings where the number of data points is largely inferior to the number of dimensions. We assume the data to lie in local and dense regions of the original space and we propose to embed each cluster into its specific ellipsoid. A new objective function is introduced, analytical solutions are derived for both the centroids and the associated ellipsoids. Furthermore, a study on the complexity of this algorithm highlights that it is of same order as the regular K-means algorithm. Results on both synthetic and real data show the efficiency of the proposed method.**

***Keywords*-clustering, feature selection, spherical k-means, information retrieval**

I. Introduction

Data clustering is the task of partitioning a set of examples $\mathbf{X}$, the data, into K sub-classes. Among others, K-means is a classical algorithm which constructs a partition by computing K centroids minimizing the overall intra-cluster dissimilarity. In the case of text clustering, due to the high dimensionality of the data, the sparsity of the *bag of words* representation as well as the specificity of textual features [1], the original K-means algorithm performs poorly. [2] have introduced the spherical K-means. The authors propose to employ the cosine similarity measure for comparing documents' bags of words instead of the classical Euclidean distance. Their study shows that their proposal yields a partitioning of the unit hyper-sphere and that it performs well when clustering high dimensional and very sparse data (more than 0.98 on average). Here, *sparsity* is defined as the ratio between the number of features equal to zero in a document and the total number of dimensions.

When the clustering problem is ill-conditioned, for example when the number n of instances is largely inferior to the number m of dimensions, feature selection can be used as a mean for obtaining a good compromise between bias and variance [3]. In the case of clustering, this results in more stable partitions, less dependent both on the noise in data and on the random initialization step. Moreover, by inhibiting the effect of non-relevant features, this process leads to more interpretable partitions and prevents the *curse of dimensionality*. Examples of settings where $n \ll m$ include micro-array data or regarding texts, the dynamical clustering of time evolving data. Consider a problem where the objective is to build a K-partition of a set of n sources described by the documents they produce over time (for example RSS-feeds, blogs or users generating content over the Internet). At each time step, the current partition must be updated with newly arrived documents and the input space (composed of the textual descriptors observed so far) is extended with all newly observed descriptors. Data sources are represented with high dimensional sparse vectors in the input space $\mathcal{X}$ and the problem rapidly falls in the setting $n \ll m$. Of course, in the more traditional task of document clustering, it may also occur that only few documents are available. Even though in this paper we do not make the distinction between these two tasks, our proposal is mainly motivated by the former one.

In the task of clustering high dimensional and very sparse data, we propose to address settings where $n \ll m$ by extending the spherical K-means algorithm for performing feature selection. More specifically, we propose to cluster on ellipsoids rather than on the unit hyper-sphere. Our main motivation is to bias the similarity measure towards the most relevant dimensions: we characterize an ellipsoid by a vector of positive weights whose effect is to dilate or contract dimensions based on their relevance for the clustering. While the weights may be set based on prior knowledge on the data, we also propose an update rule for computing the ellipsoids which maximize the overall intra-cluster similarity. Furthermore, we make the hypothesis that clusters lie in local and dense regions of the input space and we embed each cluster in its specific ellipsoid. We introduce the ellipsoidal K-means algorithm (*ellkm*) for performing feature selection in document clustering.

This paper is structured as follows: in Section II we review related works. In Section III, we introduce the proposed algorithm: we first formulate the problem and study its solution, then we analyze the convergence and complexity of the proposed algorithm. A tuning parameter s controls the shape of the ellipsoids, in Section IV we present a procedure for its automatic selection. Results on both synthetic and real data are presented in Section V. Finally, conclusion and future work are presented in Section VI.

II. Related work

Feature selection is the process of seeking an embedding of the data in which cluster structures are more easily

identified[1]. It differs from feature extraction which consists in extracting from $\mathcal{X}$ new features (also known as topics) defining a sub-space of reduced dimensionality. For instance Latent Semantic Indexing *LSI* [4] and its probabilistic variant *pLSI* [5] are classical feature extraction methods. *LSI* consists in seeking an l-dimensional sub-space (where l is user specified) describing most of the most variance in the data. The dimensions of the new space form topics expressed as linear combination of the original dimensions. Generally $l \ll m$ and the new space yields a dense representation of the data. Various methods achieve a similar goal under different frameworks and hypotheses. Conceptually, transposing the data matrix and running for example the K-means algorithm amounts to the identification of topics viewed as centroids in the document space. However, in its classical form feature extraction is not intended to inhibit the effect of certain features in favor of others. Moreover, topics require a supplementary step for interpretation and the final partition can become rather complex to explain.

In document clustering, a simple example of feature selection is the necessary pre-processing step of removing stop words and most frequent words. Recently, a framework for feature selection in clustering has been proposed [6]. For the K-means algorithm in particular, it consists in extending the objective function with a lasso type penalty: a vector of positive weights over the input space is subject to an $L1$ constraint. Here the advantage is to make use of the constraint for setting to zero irrelevant features. Even though $L1$ optimization might require a fair amount of computation, a certain advantage resides in its nullifying properties.

The aforementioned methods perform selection on the whole data in which case $\mathcal{X}$ is mapped to $\mathcal{X}'$ of reduced dimensions. Differently, our proposal is motivated by the hypothesis that cluster structures are more easily identifiable in local and dense regions of the original space and it seeks sub-spaces specifics to each cluster. In [7] the authors propose an iterative algorithm for selecting features relevant to each centroid based on their correlation measures. This method performs an in-depth search of possible solutions and therefore, it remains highly expensive and intractable on high dimensional data. Another approach consists in defining a set of weights specific to each cluster and to optimize the objective function with respect to shape constraints over the vectors of weights: the *cosa* algorithm [8] penalizes the K-means' objective function with the entropy of each of the weights vectors. The authors derive a similarity matrix from the solution of the extended objective, which can for instance be employed with a hierarchical clustering algorithm. Very similarly, the *ewkm* algorithm [9] as well as the *lac* proposal [10] explicitly extend the K-means algorithm with entropy constraints over vectors of weights local to each centroid.

These methods extend the K-means algorithm under the Euclidean distance which performs poorly on text data [1]. A more suitable approach is proposed in [11] where the authors extend the *lac* algorithm with latent semantic kernels. However this method relies on much computation and, as presented by the authors, requires external resources. Furthermore as noted earlier, feature extraction inhibits sparsity. In [12], the authors make use of the cosine similarity measure and seek cluster structures in local regions by performing feature selection on medoids rather than on centroids. However the induced algorithm introduces further user specified parameters and strongly depends on fine tuning. Moreover, as compared to the regular K-means algorithm it suffers from additional complexity.

We propose an extension of the spherical K-means algorithm for performing feature selection on text data: our proposal seeks sub-spaces of the unit hyper-sphere and it remains in line with the principle of both *cosa* and *ewkm*. To the best of our knowledge the only attempt at extending the spherical K-means algorithm for performing feature selection is [13]: the authors address the task of data fusion with the K-means algorithm under any convex distance function. To perform feature space selection they introduce a weighted objective function where a vector of positive weights adjust the influence of each representation. Even though feature space selection can be seen as an extension to feature selection, the authors do not propose an automatic update rule for deriving the weights.

III. Ellipsoidal K-means

In the following, $\boldsymbol{x}$ and $\boldsymbol{z}$ denote m-dimensional column vectors, their transpose is noted $\boldsymbol{x}^\intercal$ and $\circ$ is the Hadamard product. Scalars are non-bold, the j^{th} component of vector $\boldsymbol{x}$ is noted x_j. Furthermore the vector of all ones is noted $\mathbf{1}$ and the vector $(x_1^a, \ldots, x_m^a)$ is noted $\boldsymbol{x}^a$. Matrices are in capital bold, for example $\mathbf{I}$ is the identity matrix. The operator $\|.\|$ represents the Euclidean norm.

A. A reminder on clustering on the unit hyper-sphere

Given a set of n documents represented as non-negative bag of words vectors $\boldsymbol{x} \in \mathcal{X}$ to partition into K clusters π_k, the spherical K-means algorithm (*spkm*) [2] computes a partition maximizing the overall intra-cluster cosine-similarity: for a given cluster π_k, the centroid $\boldsymbol{c}_k$ is the point on the unit hyper-sphere which minimizes the angle formed with every data point in π_k. Formally, let $C = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$ and $\Pi = \{\pi_1, \ldots, \pi_K\}$ be a partition, when $\mathcal{X}$ is the non-negative orthant of the unit hyper-sphere, the *spkm* algorithm optimizes the following objective function over the set of

[1]In this paper, the terms embedding and sub-space are taken in the broad sense, they also refer to spaces where some of the dimensions are reduced near zero.

centroids C and the partitioning Π:

$$F_{\text{spkm}}(C,\Pi) = \sum_{k=1}^{K} \sum_{\boldsymbol{x}\in\pi_k} \boldsymbol{x}^\intercal \boldsymbol{c}_k \quad \text{s.t} \quad \forall k, \|\boldsymbol{c}_k\| = 1 \tag{1}$$

A local maximum is reached at (C^*, Π^*) defined by:

$$\boldsymbol{c}_k^* = \bar{\boldsymbol{x}}/\|\bar{\boldsymbol{x}}\|$$
$$\pi_k^* = \{\boldsymbol{x} | k = \text{argmax}_{l=1}^{K}\ \boldsymbol{x}^\intercal \boldsymbol{c}_l\}$$

for all k in $[1..K]$, where $\bar{\boldsymbol{x}}_k = \sum_{\boldsymbol{x}\in\pi_k} \boldsymbol{x}/|\pi_k|$ is equivalent to the centroid produced by the K-means algorithm under the Euclidean distance. Notice that $\boldsymbol{c}_k^*$ accounts for the projection of this centroid on the unit hyper-sphere, and that it does not explicitly depends on $|\pi_k|$ anymore.

The algorithm performs successive iterations over Π and C until a satisfactory solution is found. Because the cosine similarity treats every dimension equally, we refer to *spkm* as being full dimensional.

B. Clustering on ellipsoids

1) Principle: Making the hypothesis that clusters lie in dense regions of the original space, one may dilate or contract the dimensions with respect to their value in assessing the similitude between data points. Suppose we are given a vector of positive weights $\boldsymbol{\lambda} \in]0,1]^m$ such that $\mathbf{1}^\intercal\boldsymbol{\lambda} = 1$ and whose components assess the relevance of every dimension.

Let $\tilde{\boldsymbol{x}} = \boldsymbol{\lambda} \circ \boldsymbol{x}$ be the weighted version of a data point $\boldsymbol{x}$ on the unit hyper-sphere, that is its so-called projection and consider the ellipsoid $\mathcal{E}_{\boldsymbol{\lambda}} = \{\boldsymbol{z} \mid \|\boldsymbol{\lambda}^{-1} \circ \boldsymbol{z}\| = 1\}$. $\mathcal{E}_{\boldsymbol{\lambda}}$ is centered at the origin and its axes λ_j coincide with the axes of the original space. For every $\boldsymbol{x}$ on the unit hyper-sphere, we have $\|\boldsymbol{\lambda}^{-\mathbf{1}} \circ \boldsymbol{\lambda} \circ \boldsymbol{x}\| = \|\boldsymbol{x}\| = 1$ which implies that its projection $\tilde{\boldsymbol{x}}$ lies on $\mathcal{E}_{\boldsymbol{\lambda}}$. Therefore $\boldsymbol{\lambda}$ defines a transformation which changes the unit hyper-sphere into the ellipsoid $\mathcal{E}_{\boldsymbol{\lambda}}$.

Furthermore, given $\boldsymbol{x}$ on the unit hyper-sphere, let us measure its similarity with its projection $\tilde{\boldsymbol{x}}$, it holds that $\tilde{\boldsymbol{x}}^\intercal\boldsymbol{x} = \|\tilde{\boldsymbol{x}}\|\|\boldsymbol{x}\|\cos\alpha = \|\tilde{\boldsymbol{x}}\|\cos\alpha$ where α is the angle between $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$. It follows that the computation of $\tilde{\boldsymbol{x}}$ can be expressed as a transformation composed of a scaling $\|\tilde{\boldsymbol{x}}\|\mathbf{I}$ followed by a rotation $\mathbf{R}$ of angle α: we may note $\tilde{\boldsymbol{x}} = (\mathbf{R}(\|\tilde{\boldsymbol{x}}\|\mathbf{I}))\boldsymbol{x}$. In particular, when for all j, $\lambda_j = 1/m$ then $\tilde{\boldsymbol{x}} = \boldsymbol{x}/m$ and $\mathbf{R}$ reduces to the identity matrix $\mathbf{I}$. In this case, $\mathcal{E}_{\boldsymbol{\lambda}}$ is a sphere and the effect of the projection is to shrink $\boldsymbol{x}$ in all directions by a factor $1/m$. In the general case, the shrinking factor is influenced by the amount of information contained in both $\boldsymbol{x}$ and $\boldsymbol{\lambda}$: if the j^{th} component of $\boldsymbol{\lambda}$ holds the maximum weight then data points for which the j^{th} component is the less informative feature will be shrunken the most. Inversely, data points for which the j^{th} component is the most informative will be the most preserved. Correspondingly, the effect of the rotation factor is emphasized for data points for which the j^{th} component is the most informative as they are pivoted the closest towards the j^{th} axis.

2) Assessing similarity on an ellipsoid: While the dot product similarity yields a characterization of the angle between two vectors for two points on the unit hyper-sphere, on an ellipsoid its semantic differs. Let $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{z}}$ be two points from $\mathcal{E}_{\boldsymbol{\lambda}}$, we define the following similarity measure on the ellispoid:

$$\tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{z}} = \|\tilde{\boldsymbol{x}}\|\|\tilde{\boldsymbol{z}}\|\cos\tilde{\alpha} \tag{2}$$

where $\tilde{\alpha}$ is the angle between $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{z}}$. Here not only the angle they form but also the norms of $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{z}}$ influence their similarity measure: fixing $\tilde{\boldsymbol{x}}$ and allowing $\tilde{\boldsymbol{z}}$ to freely move on $\mathcal{E}_{\boldsymbol{\lambda}}$ one can see that the smaller $\tilde{\alpha}$ or the greater $\|\tilde{\boldsymbol{z}}\|$, the more similar is $\tilde{\boldsymbol{z}}$ to $\tilde{\boldsymbol{x}}$. Now because data points on ellipsoids have their norm constrained, the dot product between two points from $\mathcal{E}_{\boldsymbol{\lambda}}$ measures a compromise between both their norms and the angle they form. Depending on their location on the ellipsoid, the similarity between two points might be more sensitive in their norms or in the angle they form.

We now give an upper bound on the similarity between two points $\tilde{\boldsymbol{z}}$ and $\tilde{\boldsymbol{x}}$ on the ellipsoid $\mathcal{E}_{\boldsymbol{\lambda}^{\frac{1}{2}}}$: it holds that $\tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{z}} = (\boldsymbol{\lambda}\circ\boldsymbol{x})^\intercal\boldsymbol{z} \leq \|\boldsymbol{\lambda}\circ\boldsymbol{x}\|\|\boldsymbol{z}\|$ from the Cauchy-Schwartz inequality. Now since $\tilde{\boldsymbol{x}} = \boldsymbol{\lambda}^{\frac{1}{2}} \circ \boldsymbol{x}$ and $\boldsymbol{z}$ is on the unit hyper-sphere, i.e. $\|\boldsymbol{z}\| = 1$,

$$\|\boldsymbol{\lambda}\circ\boldsymbol{x}\|\|\boldsymbol{z}\| = \frac{(\boldsymbol{\lambda}\circ\boldsymbol{x})^\intercal(\boldsymbol{\lambda}\circ\boldsymbol{x})}{\|\boldsymbol{\lambda}\circ\boldsymbol{x}\|} = \frac{\tilde{\boldsymbol{x}}^\intercal(\boldsymbol{\lambda}^{\frac{3}{2}}\circ\boldsymbol{x})}{\|\boldsymbol{\lambda}\circ\boldsymbol{x}\|}$$

by noting $\tilde{\boldsymbol{c}} = \boldsymbol{\lambda}^{\frac{1}{2}} \circ \frac{\boldsymbol{\lambda}\circ\boldsymbol{x}}{\|\boldsymbol{\lambda}\circ\boldsymbol{x}\|}$ we obtain that $\tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{z}} \leq \tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{c}}$. Likewise for $\tilde{\mathbf{X}}$, a set of n points in $\mathcal{E}_{\boldsymbol{\lambda}^{\frac{1}{2}}}$ it follows that,

$$\sum_{\tilde{\boldsymbol{x}}\in\tilde{\mathbf{X}}} \tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{z}} \leq \sum_{\tilde{\boldsymbol{x}}\in\tilde{\mathbf{X}}} \tilde{\boldsymbol{x}}^\intercal\tilde{\boldsymbol{c}}, \qquad \text{for } \tilde{\boldsymbol{c}} = \boldsymbol{\lambda}^{\frac{1}{2}} \circ \frac{\boldsymbol{\lambda}\circ\bar{\boldsymbol{x}}}{\|\boldsymbol{\lambda}\circ\bar{\boldsymbol{x}}\|}$$

Therefore $\tilde{\boldsymbol{c}}$ is a centroid of $\tilde{\mathbf{X}}$ induced by the similarity measure on the ellispoid given in eq. (2).

C. Problem formulation

We propose to employ the measure of similarity on the ellipsoid as described in the previous section instead of the cosine-similarity on the unit hyper-sphere. Furthermore, in order to discover local cluster structures in the original space, we allow each cluster π_k to be associated with a specific ellipsoid fully defined by a weight vector $\boldsymbol{\lambda}_k \in [0,1]^m$: it must be noted that the ellipsoid associated to cluster π_k is now considered in the sub-space defined by all features for which a data point in cluster π_k is non zero. In the following we denote by Λ the set $\{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_K\}$. Even though the ellipsoids may be set and hold fixed by means of prior knowledge on the data, we propose to seek the ellipsoids which maximize the intra-cluster similarity.

Because it is in general easier to compute a homogeneous partition on a unique dimension, the set of solutions Λ^* is

in general trivial: it corresponds to the set of vectors with all but one component close to zero. We introduce a tuning parameter s in the range $[0, 1[$ whose effect is to control the shape of the ellipsoids: when $s = 0$ then for all k, $\boldsymbol{\lambda}_k^s = \mathbf{1}$ and $\mathcal{E}_{\boldsymbol{\lambda}_k^s}$ reduces to the unit hyper-sphere. As s tends toward 1, Λ^* degenerates into the set of trivial solutions for which all but one component are close to zero. In this sense, the problem we formulate can be seen as a generalization of the spherical case.

Given a set of n vectors in the non-negative orthant of the unit hyper-sphere $\boldsymbol{x} \in \mathcal{X}$, an integer K and a real $s \in [0, 1[$, we wish to maximize the following objective function over the set of centroids C, the set of weight vectors Λ and the partition Π:

$$F_{\text{ellkm}}(C, \Lambda, \Pi) = \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in \pi_k} \left(\boldsymbol{\lambda}_k^{\frac{s}{2}} \circ \boldsymbol{x}\right)^\intercal \left(\boldsymbol{\lambda}_k^{\frac{s}{2}} \circ \boldsymbol{c}_k\right) \quad (3)$$

$$s.t \begin{cases} \forall k, & \mathbf{1}^\intercal \boldsymbol{\lambda}_k = 1 \\ \forall k, & \|\boldsymbol{c}_k\| = 1 \end{cases}$$

Theorem 1 states the solution for (3). When s is zero, (3) reduces to (1) and the induced similarity measure is the full dimensional cosine similarity. A larger s amounts for much weight on dimensions participating strongly in the similitude between data points and their centroids. Because weights' components sum to one, non-relevant dimensions are pushed toward zero.

Theorem 1: F_{ellkm} reaches a local maximum at (C^*, Λ^*, Π^*) defined by:

$$\boldsymbol{c}_k^* = \frac{\boldsymbol{\lambda}_k^s \circ \bar{\boldsymbol{x}}_k}{\|\boldsymbol{\lambda}_k^s \circ \bar{\boldsymbol{x}}_k\|} \quad (4)$$

$$\boldsymbol{\lambda}_k^* = \frac{(\bar{\boldsymbol{x}}_k \circ \boldsymbol{c}_k)^{\frac{1}{1-s}}}{\mathbf{1}^\intercal (\bar{\boldsymbol{x}}_k \circ \boldsymbol{c}_k)^{\frac{1}{1-s}}} \quad (5)$$

$$\pi_k^* = \{\boldsymbol{x} | k = \text{argmax}_{l=1}^{K} \ (\boldsymbol{\lambda}_l^{\frac{s}{2}} \circ \boldsymbol{x})^\intercal (\boldsymbol{\lambda}_l^{\frac{s}{2}} \circ \boldsymbol{c}_l)\} \quad (6)$$

Furthermore, $\forall s \in [0, 1[$, C^* is a global maximum of F_{ellkm} over C and Λ^* is a global maximum of F_{ellkm} over Λ.

Proof: Holding Λ fixed, and considering the ellipsoid $\mathcal{E}_{\boldsymbol{\lambda}^{\frac{s}{2}}}$, it follows from the expression of the centroid on an ellipsoid (Section III-B) that $\tilde{\boldsymbol{c}}_k^* = \boldsymbol{\lambda}^{\frac{s}{2}} \circ \frac{\boldsymbol{\lambda}_k^s \circ \bar{\boldsymbol{x}}_k}{\|\boldsymbol{\lambda}_k^s \circ \bar{\boldsymbol{x}}_k\|} = \boldsymbol{\lambda}^{\frac{s}{2}} \circ \boldsymbol{c}_k^*$. From an analytical point of view, the same result is achieved (proof omitted) by setting to 0 the derivative of the Lagrangian of (3) over C.

Furthermore, holding C fixed, and denoting γ_k the Lagrange multipliers, the Lagrangian of (3) over Λ is:

$$L(\Lambda) = \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in \pi_k} \sum_{j=1}^{m} \lambda_{kj}^s x_j c_{kj} - \left[\sum_{k=1}^{K} \gamma_k \left(\sum_{j=1}^{m} \lambda_{kj} - 1\right)\right]$$

The derivatives of L for the j^{th} component of the k^{th} cluster and for the γ_k multiplier are then:

$$\frac{\partial L}{\partial \lambda_{kj}} = \sum_{\boldsymbol{x} \in \pi_k} s \lambda_{kj}^{s-1} x_j c_{kj} - \gamma_k \quad (7)$$

$$\frac{\partial L}{\partial \gamma_k} = 1 - \sum_{j=1}^{m} \lambda_{kj} \quad (8)$$

Setting (7) to zero leads to, $\forall s \neq 1$:

$$\lambda_{kj} = \left[\frac{\gamma_k}{\sum_{\boldsymbol{x} \in \pi_k} x_j c_{kj} s}\right]^{\frac{1}{s-1}}$$

Moreover setting (8) to zero:

$$\gamma_k = s \left[\sum_{l=1}^{m} \frac{1}{\left[\sum_{\boldsymbol{x} \in \pi_k} x_l c_{kl}\right]^{\frac{1}{(s-1)}}}\right]^{(1-s)}$$

Finally we obtain,

$$\lambda_{kj} = \frac{1}{\sum_{l=1}^{m} \left[\frac{\sum_{\boldsymbol{x} \in \pi_k} x_j c_{kj}}{\sum_{\boldsymbol{x} \in \pi_k} x_l c_{kl}}\right]^{\frac{1}{s-1}}}$$

We notice that $\boldsymbol{\lambda}_k$ is expressed as the Hadamard product between $\bar{\boldsymbol{x}}$ and $\boldsymbol{c}_k$ to the power $1/(1-s)$ and normalized by their scalar product, leading to eq. (5) when expressed for the whole vector.

Finally, L being concave over Λ for all s in $[0, 1]$ and Λ^* being undefined for $s = 1$, optimality is then guaranteed $\forall s \in [0, 1[$. We note that as $s \to 1$, $\boldsymbol{\lambda}_k^*$ degenerates into a trivial solution with all components but one close to zero. ■

D. Proposed algorithm

Algorithm 1 is our proposed ellipsoidal K-means algorithm. With respect to the spherical K-means algorithm, a new step is added for computing the ellipsoids in which the current centroids are embedded (line 14). At initialization step, centroids are randomly drawn and ellipsoids are set to the sphere of radius $1/m$. Update formulas are provided in eq. (4-6) of Theorem 1. The algorithm stops when either the maximum number of iterations $\max_T$ is reached or when the improvement is lower than a predefined threshold η (in our experiments η is empirically set to 10^{-8}).

E. Convergence

We demonstrate the convergence of Algorithm 1 by first showing that each step increases the value of F_{ellkm}.

Let F_t be the value of the objective function F_{ellkm} at step t of Algorithm 1. Let Π_t be the partition, C_t the set of centroids and Λ_t the set of weight vectors at step t. According to Theorem 1,

$$\begin{aligned} F_t = F_{\text{ellkm}}(C_t, \Lambda_t, \Pi_t) &\leq F_{\text{ellkm}}(C_t^*, \Lambda_t^*, \Pi_t) \\ &= F_{\text{ellkm}}(C_{t+1}, \Lambda_{t+1}, \Pi_t) \end{aligned}$$

Algorithm 1 *Ellipsoidal K-means*

1: *input:* $\mathbf{X}, K, s$
2: *output:* (Π, C)
3: $C_0 \leftarrow K$ random centroids
4: $\Lambda_0 \leftarrow K$ uniform weight vectors set to $(\frac{1}{m}, \ldots, \frac{1}{m})$
5: Π_0 is the initial random partition
6: $t \leftarrow 0$
7: **while** $t < \max_T$ **and** $\Delta F > \eta$ **do**
8: **for all** $\boldsymbol{x} \in \mathcal{X}$ **do**
9: $l = \text{argmax}_{k=1}^{K} \left(\boldsymbol{\lambda}_k^{\frac{s}{2}} \circ \boldsymbol{x}\right)^{\mathsf{T}} \left(\boldsymbol{\lambda}_k^{\frac{s}{2}} \circ \boldsymbol{c}_k\right)$
10: update π_l with $\boldsymbol{x}$
11: **end for**
12: **for all** $k \in [1..K]$ **do**
13: update $\boldsymbol{c}_k$ (according to eq. (4))
14: update $\boldsymbol{\lambda}_k$ (according to eq. (5))
15: **end for**
16: $C_{t+1} \leftarrow \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K\}$
17: $\Lambda_{t+1} \leftarrow \{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_K\}$
18: $\Pi_{t+1} \leftarrow \{\pi_1, \ldots, \pi_K\}$
19: $\Delta F = F_{\text{ellkm}}(C_{t+1}, \Lambda_{t+1}, \Pi_{t+1}) - F_{\text{ellkm}}(C_t, \Lambda_t, \Pi_t)$
20: $t \leftarrow t + 1$
21: **end while**

Furthermore, by definition of Π^* we have,

$$\begin{aligned} F_{\text{ellkm}}(C_{t+1}, \Lambda_{t+1}, \Pi_t) &\leq F_{\text{ellkm}}(C_{t+1}, \Lambda_{t+1}, \Pi_t^*) \\ &= F_{\text{ellkm}}(C_{t+1}, \Lambda_{t+1}, \Pi_{t+1}) = F_{t+1} \end{aligned}$$

Therefore,

$$F_t \leq F_{t+1}$$

The convergence of the algorithm is finally ensured by observing that F_{ellkm} is always upper bounded by a positive constant and therefore ΔF tends towards zero as $\max_T$ tends towards infinity.

F. Complexity

In Algorithm 1, the reassignment step (line. 9) leading to the update of Π involves the computation of the complete similarity matrix between the data points and the centroids, this step performs nKm operations. Additionally, the computation of the new centroids and the new ellipsoids both involve nm operations. In the worst case, the algorithm performs $\max_T$ iterations, therefore the overall complexity is $O(nKm \times \max_T)$ which is of same order as the regular K-means algorithm.

IV. Selection of the tuning parameter s

A. Semantic of the parameter s

In Section III, we introduced a tuning parameter s for controlling the shape of the ellipsoid $\mathcal{E}_{\boldsymbol{\lambda}_k^{\frac{s}{2}}}$. For small s, the ellipsoids approach the form of a sphere thus inhibiting the influence of the weights $\boldsymbol{\lambda}_k$. When s is closer to 1, the ellipsoids tend towards straight lines, that is $\boldsymbol{\lambda}_k$ is the vector of all but one component close to zero: the axis which holds the most similitude between data points and their centroids receives all the weight. Therefore, in cases where it is aimed to take account of a high number of dimensions, a small s is most suited, while in other cases, where the clusters lie in extremely local and dense regions of the original space, a large s is preferred. It must also be noted that in Algorithm 1, a large s tends to produce drastic changes at each iteration while a small s tends to produce smoother changes. As a result s also influences the stability of the partition. In this sense s is akin to a learning rate parameter.

Because in clustering, little information is known about the data, we describe a general procedure for automatically tuning s.

B. Procedure for automatic selection

The parameter s may be tuned in a qualitative way: an initial guess based on prior knowledge is successively refined until the algorithm produces satisfying partitions. Similarly, a natural procedure consists in evaluating the overall intra-cluster similarity for different values of s and choosing the value that yields the most homogeneous clusters. However on ellipsoids $\mathcal{E}_{\boldsymbol{\lambda}^{\frac{s}{2}}}$, the similarity measure directly depends on the parameter s. As a consequence, changing s amounts to changing the objective function itself and one cannot compare the quality of the partitions for different values of s.

Inspired by the *elbow method* [14], the *gap statistic* method [15] has been proposed to estimate the number K of clusters in a dataset $\mathbf{X}$. This procedure consists in normalizing the objective function for eliminating the effect of K's influence over the partition.

In [6], the authors propose to extend this procedure for estimating a continuous parameter. Let $\mathbf{X}_{b \in [1..B]}$ be B random variants of $\mathbf{X}$ which can be obtained by repeatedly permuting every data point over each dimension [6]. Considering that the clustering relies on a similarity measure parametrized by s, let F denote the value of the objective function computed over $\mathbf{X}$ and F_b denotes its value over $\mathbf{X}_b$. The gap measure associated to s is defined as:

$$\text{gap}(s) = \log F - \frac{1}{B} \sum_{b=1}^{B} \log F_b$$

Given a set of user supplied values S, the gap statistic associated to each each $s \in S$ is first computed, then $s_0^* = \text{argmax}_{s \in S} \text{gap}(s)$ is selected.

Now in settings where $n \ll m$ and for methods relying on a random initialization step such as the K-means algorithm, it is important to assess the quality of the clustering as measured by gap(s) by performing different simulations over different initial centroids. While in [6] the authors make use of a majority voting scheme for aggregating each s_0^* computed over one simulation, we propose the following

heuristic for the choice of s^*:

$$s^* = \text{argmax}_{s \in S} \sum_{i=1}^{N} \text{gap}_i(s) - \tau_s \quad (9)$$

where τ_s is the standard-deviation of $\text{gap}_{i \in [1..N]}(s)$ and $\text{gap}_i(s)$ is computed over one random initialization of the centroids. When N simulations are performed, the choice of s^* is less sensitive to extreme values of gap_i than different choices of s_0^*, aggregated over a majority voting scheme. It must be noted that the gap measure relies on the hypothesis that clusters are well separated in the input space. When clusters do not exhibit such behavior, for example in the presence of noise or when clusters are under represented, the gap procedure seeks the embedding that holds the most homogeneous partition of $\mathbf{X}$ compared to $\mathbf{X}_{b \in [1..B]}$. Our proposal requires the partitions to be the most stable as well. If $s^* = 0$ is selected in (9), it means that no sub-space leads to a better partition than the full dimensional original space. The complete procedure for selecting s is described as follows:

1) Generate B datasets $\mathbf{X}_b$ by randomly permuting the components of each data point in $\mathbf{X}$. In the case of texts, this amounts to randomly swapping words between documents, thus breaking any cluster structure that may exist.
2) For each $s \in S$ including the spherical case ($s = 0$), compute the overall intra-cluster similarity on the original dataset and on the B random datasets. Repeat for N different random initializations.
3) For each $s \in S$, evaluate its gap measures: compute $\text{gap}_i(s)$ for the i^{th} simulation. Finally compute τ_s as the standard-deviation of $\text{gap}_{i \in [1..N]}(s)$.
4) Choose s^* as defined in eq. (9).

Referring to Section III-F, the complexity of this procedure is $O(|S|(B+1) \times nKm \times \text{step}_{\text{max}})$ where $|\mathcal{S}|$ is the number of values tested for fitting s^*.

V. Experiments and results

We present results obtained on a set of synthetic data, then we study the performance of *ellkm* on the standard 20-newsgroup dataset [16].

A. Synthetic data

1) Data generation: We first evaluate *ellkm* on a synthetic corpus composed of $K = 3$ clusters. The data simulate a corpus of n documents on $M = 3000$ words weighted in the range $[0, 1]$. The corresponding bag of words vectors have an average sparsity of 0.98. All clusters are of equal size n/K.

To account for clusters' specific vocabulary, each cluster π_k is assigned a set of 100 specific words. The generation of a document $\boldsymbol{x}$ is as follow:

- A sparsity degree q is drawn from the normal distribution with mean 0.98 and standard-deviation 10^{-2}, the number of non-zero components of $\boldsymbol{x}$ is set to $p = M(1 - q)$
- p non-zero components are set to a value uniformly drawn in the range $[\epsilon, 1]$ where ϵ is a small positive constant, set to 10^{-3}. 40% of them are randomly sampled in the set of features specific to k, the remaining are sampled in non-specific features and represent 60% of noise in the data.

Table I
Description of the 10 synthetic datasets.

dataset	n	m	n / m	sparsity
1	30	1088	0.027	0.98
2	60	1736	0.035	0.98
3	90	2239	0.04	0.98
4	120	2514	0.047	0.98
5	150	2608	0.057	0.98
6	180	2756	0.065	0.98
7	210	2799	0.075	0.98
8	240	2879	0.083	0.98
9	270	2923	0.092	0.98
10	300	2941	0.1	0.98

2) Datasets: We generate different datasets by varying the number $n \in [30, 300]$ of generated documents. The dimensionality m of each dataset is then given as the overall number of non-zero features in the data. For low n, clusters are highly under represented, that is $n \ll m$. As n increases, so does the ratio between the number of instances n and the dimensionality m. Table I reports the quantity m and the ratio n/m for each dataset.

3) Settings and protocol: We compare the ellipsoidal K-means proposal with the spherical K-means algorithm. For each run, the tuning parameter s is selected with the procedure described in Section IV-B: $B = 10$ reference datasets are generated and 10 different values for s are tested in the range from 0 to 0.5 (in experiments not reported here, we observed that values greater than 0.5 do not produce meaningful partitions).

Both systems are evaluated in a supervised manner: we use the Normalized Mutual Information measure [17] to assess the quality of the partitions:

$$\text{nmi}(\Pi, Y) = \frac{I(\Pi, Y)}{\sqrt{H(\Pi)H(Y)}}$$

where Y is the vector of the documents' true labels, Π is the partition being evaluated, $I(\Pi, Y)$ is the mutual information measure of Π and Y; $H(\Pi)$ and $H(Y)$ are respectively the entropy measure of Π and Y. For this normalized variant, $\text{nmi}(\Pi, Y) \in [0, 1]$ must be maximized.

For each setting, 20 runs are performed and both systems are initialized with the same initial random partition.

4) Results: In the upper left part of Figure 1 we compare the average performance of the automatic parameter selection procedure (s^*) with the spherical case ($s = 0$).

When n/m is low, we observe that our proposal produces better partitions than the full dimensional spherical K-

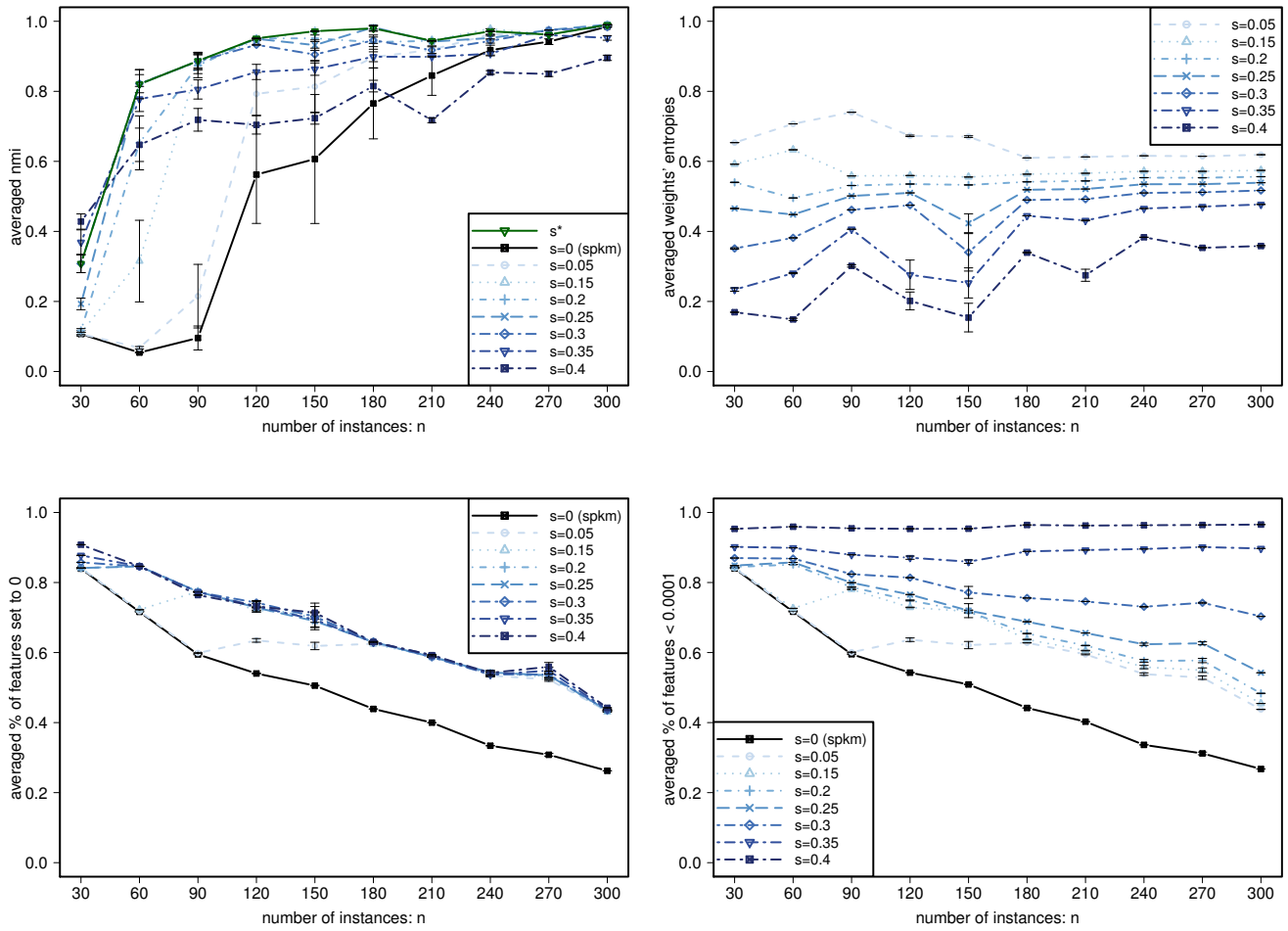

Figure 1. Average performance measures with variances over 20 runs for 8 settings. The x axis is the number of instances, the y axis represents the average nmi (upper left), the entropies of the weights vectors (upper right), the percentage of centroid's components set to zero (lower left) and the percentage of centroids' components inferior to 10^{-3} (lower right).

means algorithm. More specifically, when $n \ll m$ the true distribution lies on fewer dimensions, while in the original space there is not enough data points to recover features specific to each cluster, ellipsoidal K-means embeds them in a sub-space where it is easier to recover their underlying structure. In this setting, feature selection allows one to account for the most relevant features in the data. In contrast, as indicated by the drop in performance for $s = 0.4$, higher ratios require more features (each of the 100 features specific to every cluster) to recover the initial distribution of the data. We observe that while both *spkm* and *ellkm* handle well the latter setting, *spkm* undergoes a large drop in performance for the former setting.

When clusters are under-represented, we notice that the partitions produced by *ellkm* are more stable as indicated by the variance of the *nmi* measure. Finally we observe that the procedure for selecting the parameter s systematically chooses the most relevant ellipsoids, except in the first dataset ($n = 30$) where we suspect that the noise in the data interferes with the gap measure.

The lower left part of Figure 1 shows the percentage of centroids' components set to zero. As expected, *ellkm* promotes sparser centroids than *spkm*, as a result the induced partitions are expected to remain more stable. Nevertheless, we observe that while *ellkm* pushes irrelevant features towards zero, it does not set every irrelevant feature to exactly zero. Indeed, even though a significant number of components are set to zero with respect to *spkm*, the effect is not emphasized with increasing s. However, on the lower right part of Figure 1 we plotted the percentage of centroids' component strictly inferior to $\epsilon = 10^{-3}$. Here we observe that as s increases, so does the number of near zero components.

In a similar way, on the upper right part of Figure 1,

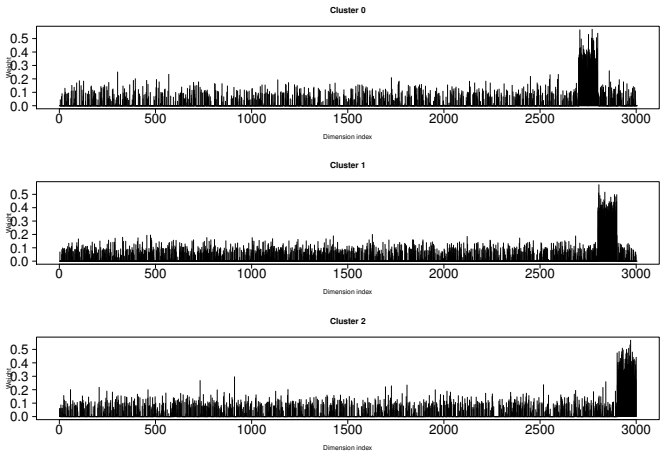

Figure 2. Examples of weights $\boldsymbol{\lambda}_k$ associated with each centroid and computed for $s^* = 0.2$ on the synthetic dataset of size $n = 120$.

the average entropy of the weight vectors $\boldsymbol{\lambda}_k$ decreases when s increases. A noticeable drop in entropy is observed for $n = 150$ and higher s. As each dataset is generated independently from the others and contains much noise, here we suspect that the algorithm gets stuck on few features.

Figure 2 shows the vectors of weights associated to the 3 final centroids for $n = 120$ and $s^* = 0.2$. We observe that a large weight was correctly assigned to each of the 100 specific features.

B. 20-newsgroup data

In this section, experiments are reported on real data. Three different corpora have been built by sampling 6 categories from the *20-newsgroup* data, namely: *soc.religion*, *comp.graphics*, *rec.sport.baseball*, *sci.space*, *talk.politics.guns*, *talk.politics.mideast*. For each corpus, two datasets are extracted: the first one contains very few documents so that $n \ll m$, the second one is composed of much more documents. A description of the datasets is provided in Table II. The K categories composing each dataset are all of equal size: they all contain n/K documents.

1) Data pre-processing: Document headers as well as all formatting content are first removed. All words are then put to lowercase and lemmatized. Finally words are filtered based on their part of speech information: all auxiliary verbs (e.g *be, have*), determiners (e.g *the, an*) as well as conjunctions (e.g *or, but*) are discarded. Both lemmatization and part of speech tagging is performed with TreeTagger [18]. We adopt the classical *tf/idf* weighting scheme which have the advantage of penalizing words occurring too frequently in the corpus. Lastly, terms which occur in more than 20% of the documents as well as those which occur in less than 2 documents are discarded; documents shorter than 10 words are discarded. The formatted dataset $\mathbf{X}$ is projected onto the unit hyper-sphere.

2) Settings: As detailed below, we compare *ellkm* with 4 other algorithms: *sparcl* [6] and *ewkm* [9] are both extensions of the standard K-means (under the Euclidean distance) and both perform feature selection. While *ewkm* derives subspaces specific to each cluster, *sparcl* seeks an embedding for the overall data. *spkm* [2] and *plsa* [3] are two classical methods for clustering high dimensional and sparse data, they both work in the full dimensional setting. Each system is tuned as follows:

sparcl includes a gap procedure for selecting the tuning parameter which controls the dimensionality of the embedding sub-space [6]. In our experiments, a range of 10 values over $B = 10$ reference datasets is tested for each dataset.

ewkm employs a tuning parameter γ for controlling the entropy of the vectors of weights defining the clusters' sub-spaces. In their experiments, as the authors manually set γ [9], we evaluate 10 different values and we keep the one holding the best performance in terms of *nmi* score. It must be noted that this tuning procedure takes account of the true labels and therefore, *ewkm* is put at an advantage.

spkm, even though our proposal extends the spherical case, we wish to compare our results with those produced in the full dimensional setting.

plsa is another classical method in information retrieval for high dimensional and very sparse data [5]. Here, we view *plsa* as a full dimensional clustering algorithm, in our experiments, document $\boldsymbol{x}$ is associated with cluster $\pi_k = \text{argmax}_{k=1}^{K} p(\boldsymbol{z}_k|\boldsymbol{x})$, where topic $\boldsymbol{z}$ is viewed as a centroid.

For each system, the number of centroids is set to the number of true classes.

3) Protocol: We perform 20 runs for each system. As in Section V-A, partitions are evaluated with the *nmi* measure. We also use the standard *Rand index* [19] as well as the dominant labels' frequencies averaged over the whole partition also known as the *purity score*. While the *nmi* metric measures the dependence between the partition and the true classes, the *purity* assesses the soundness of the partition, that is, how pure are the clusters. As for the *Rand index*, it is similar to the *accuracy* in supervised setting, it can be viewed as a measure of matching between two partitions.

4) Results: Table III reports results over small datasets ($n \ll m$) and results over larger datasets are given in Table IV. N/A's in the tables stand for results not provided in 24 hours.

Both *sparcl* and *ewkm* which rely on the Euclidean distance encounter difficulties on the three corpora. As pointed out in Section II, these results confirms the inadequacy of the K-means algorithm under the Euclidean distance for such data. On the contrary, we observe that both *spkm* and *plsa* which rely on a full dimensional similarity measure obtain fairer results.

In datasets 1.1 and 3.1 of Table III which are composed of

Table II
DESCRIPTION OF THE 3 CORPORA EXTRACTED FROM THE 20-NEWSGROUP DATA. TWO DATASETS ARE BUILT FROM EACH CORPUS: 1.1, 2.1 AND 3.1 REFER TO $n << m$ DESIGNS WHILE 1.2, 2.2 AND 3.2 REFER TO TRADITIONAL DESIGNS.

dataset	categories	n	m	n/m	sparsity
1.1	soc.religion/comp.graphics	272	2455	0.1	0.98
2.1	comp.graphics/rec.sport.baseball/sci.space	250	1699	0.1	0.98
3.1	talk.politics.guns/talk.politics.mideast	260	3164	0.1	0.98
1.2	soc.religion/comp.graphics	1772	8895	0.2	0.99
2.2	comp.graphics/rec.sport.baseball/sci.space	2574	10368	0.2	0.99
3.2	talk.politics.guns/talk.politics.mideast	1790	10712	0.2	0.99

Table III
COMPARISONS ON 3 SMALL DATASETS ($n \ll m$) EXTRACTED FROM THE 20-NEWSGROUP DATA.

dataset	system	nmi	rand	purity
1.1	Ellkm	0.57 ± 0	0.84 ± 0	0.91 ± 0
	Spkm	0.26 ± 0.07	0.66 ± 0.02	0.72 ± 0.03
	Plsa	0.38 ± 0.04	0.73 ± 0.02	0.81 ± 0.02
	Sparcl	0.08 ± 0	0.5 ± 0	0.55 ± 0
	Ewkm	0.11 ± 0.01	0.56 ± 0	0.65 ± 0.01
2.1	Ellkm	0.36 ± 0.01	0.72 ± 0	0.7 ± 0.01
	Spkm	0.14 ± 0.02	0.62 ± 0	0.52 ± 0.02
	Plsa	0.12 ± 0.01	0.61 ± 0	0.53 ± 0.01
	Sparcl	0.17 ± 0	0.44 ± 0	0.47 ± 0
	Ewkm	0.09 ± 0	0.58 ± 0	0.5 ± 0
3.1	Ellkm	0.12 ± 0.01	0.58 ± 0	0.68 ± 0.01
	Spkm	0.02 ± 0	0.51 ± 0	0.56 ± 0
	Plsa	0.02 ± 0	0.51 ± 0	0.58 ± 0
	Sparcl	0.09 ± 0	0.5 ± 0	0.56 ± 0
	Ewkm	0.04 ± 0	0.52 ± 0	0.59 ± 0

Table IV
COMPARISONS ON 3 LARGE DATASETS EXTRACTED FROM THE 20-NEWSGROUP DATA.

dataset	system	nmi	rand	purity
1.2	Ellkm	0.76 ± 0	0.92 ± 0	0.96 ± 0
	Spkm	0.77 ± 0	0.93 ± 0	0.96 ± 0
	Plsa	0.75 ± 0	0.92 ± 0	0.96 ± 0
	Sparcl	0.15 ± 0	0.53 ± 0	0.62 ± 0
	Ewkm	0.09 ± 0.01	0.55 ± 0	0.63 ± 0.01
2.2	Ellkm	0.67 ± 0	0.88 ± 0	0.9 ± 0
	Spkm	0.67 ± 0	0.88 ± 0	0.9 ± 0
	Plsa	0.53 ± 0.01	0.81 ± 0	0.82 ± 0.01
	Sparcl	N/A	N/A	N/A
	Ewkm	0.07 ± 0	0.56 ± 0	0.47 ± 0
3.2	Ellkm	0.25 ± 0.04	0.64 ± 0.02	0.72 ± 0.02
	Spkm	0.25 ± 0.04	0.64 ± 0.02	0.72 ± 0.02
	Plsa	0.18 ± 0.03	0.61 ± 0.01	0.71 ± 0.01
	Sparcl	N/A	N/A	N/A
	Ewkm	0.03 ± 0	0.52 ± 0	0.57 ± 0

$K = 2$ categories, the Rand index as well as the purity score obtained by *sparcl* indicates that the partitions it produces remain close to random. It contrasts with the relatively better performance it obtains over the dataset 2.1 which is composed of $K = 3$ categories: it seems that on this dataset the tuning procedure finds better indication of cluster structure. On the opposite, *ewkm* achieves its best run over the dataset 1.1, however its performance remains inferior to *sparcl* over all other datasets. This may be due to the fact that *ewkm* seeks K sub-spaces but cannot estimate correctly all of its parameters in the document clustering setting.

For each system, best results are obtained on the first corpus (datasets 1.1 and 1.2). This corpus indeed exhibits two well separated clusters and except *sparcl* and *ewkm* each system tends to recover the documents' true categories. The second corpus (datasets 2.1 and 2.2) is made of three well separated categories, each of which is less populated than in the first corpus. Both *spkm* and *plsa* tend to produce accurate partitions even though they present inferior performance. The third corpus (datasets 3.1 and 3.2) however is made of two sub-classes from a common category (*politics*). Clusters are expected to share much vocabulary with one another and indeed, every system struggles on this corpus.

When $n \ll m$ (datasets 1.1, 2.1, 3.1), *ellkm* clearly performs better in all quality criteria. Furthermore, on dataset 1.1 where full dimensional systems produce meaningful results, we observe that the partitions produced by *ellkm* are also more stable.

When n is larger (datasets 1.2, 2.2, 3.2), *plsa* and *spkm* achieve equivalent results. We note that *plsa* gives poorer partitions for dataset 2.2 that can be attributed to the tuning procedure: we set the maximum number of iterations to 80, a higher value may lead to better results at the cost of running time and variance. In the larger setting, the selection procedure of *ellkm* does not find sub-spaces in which clusters are better represented than in the full dimensional input space, except for the dataset 1.2 where the value $s^* = 0.05$ is retained. As the number of documents n grows, their true categories are well described on more features and *ellkm* reduces to *spkm*. It must be noted that *sparcl* does not terminate for the dataset 2.2 and 2.3. The advantage of the lasso type penalty it employs is to set some of the features exactly to zero but at the cost of heavy computation.

Table V reports the average centroids' sparsities for both *ellkm* and *spkm*. For each dataset, *ellkm* produces sparser centroids and therefore more interpretable partitions. Also, in situations where speed matters, an efficient implementation can take benefit of components set to zero when computing similarities between pairs of objects.

VI. CONCLUSION

We proposed an ellipsoidal K-means algorithm which is an extension of the spherical K-means algorithm for feature selection in high dimensional and very sparse data. We make

Table V
COMPARISONS OF CENTROIDS' SPARSITIES FOR THE ELLIPSOIDAL K-MEANS AND THE SPHERICAL K-MEANS.

dataset	Ellkm	Spkm
1.1	0.36 ± 0.02	0.25 ± 0
1.2	0.38 ± 0	0.29 ± 0
2.1	0.52 ± 0.01	0.28 ± 0
2.2	0.39 ± 0.01	0.39 ± 0.01
3.1	0.28 ± 0.02	0.2 ± 0.02
3.2	0.21 ± 0	0.21 ± 0

the hypothesis that clusters lie in local and dense regions of the original space and we exploit a transformation which changes the unit hyper-sphere into ellipsoids. An additional step is added to the original K-means algorithm for updating the ellipsoids local to each cluster. The resulting algorithm computes both the centroids and the ellipsoids maximizing the overall intra-cluster similarity. A tuning parameter allows to control the shape of the ellipsoids: values close to 0 yield the spherical K-means algorithm while larger values inhibit the effect of less informative features. We showed the efficiency of an automatic procedure for selecting the parameter s and we adapt a new heuristic for taking into account the variance in the process.

We conducted several experiments over both synthetic and real data. In settings where the number of instances is largely inferior to the number of dimensions, the results show the efficiency of our proposal. We also observe that our extension produces simpler centroids for which more components are set to zero. Furthermore, in classical settings, our experiments show that the selection procedure successively reduces the ellipsoids to the unit hyper-sphere, therefore yielding a full dimensional algorithm.

Our work is mainly motivated by the clustering of dynamical sources of information producing documents over time: at every time step, newly seen descriptors enrich the input space, sources representations become rapidly very large and highly sparse. A perspective of this work is to study the application of our proposal in a framework for data stream analysis.

REFERENCES

[1] A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *Workshop on Artificial Intelligence for Web Search*, 2000, pp. 58–64.

[2] I. Dhillon and D. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, pp. 143–175, 2001.

[3] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, Biometrics, Ed., 2009.

[4] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse Processes*, vol. 25, no. 2, pp. 259–284, 1998.

[5] T. Hofmann, "Probabilistic latent semantic indexing," in *the Int. Conf. on Research and development in information retrieval*, 1999, pp. 50–57.

[6] D. M. Witten and R. Tibshirani, "A framework for feature selection in clustering," *Journal of the American Statistical Association*, vol. 105, no. 490, pp. 713–726, 2010.

[7] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, and J. S. Park, "Fast algorithms for projected clustering," *ACM SIGMOD Record*, vol. 28, no. 2, pp. 61–72, 1999.

[8] J. Friedman and J. Meulman, "Clustering objects on subsets of attributes (with discussion)," *Journal of the Royal Statistical Society*, vol. 66, no. 4, pp. 815–849, 2004.

[9] L. Jing, M. Ng, and J. Huang, "An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, pp. 1026–1041, 2007.

[10] C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, and D. Papadopoulos, "Locally adaptive metrics for clustering high dimensional data," *Data Mining and Knowledge Discovery*, vol. 14, pp. 63–97, 2007.

[11] P. Wang and C. Domeniconi, "Building semantic kernels for text classification using wikipedia," in *the Int. Conf. on Knowledge discovery and data mining*, 2008, pp. 713–721.

[12] A. Kalogeratos and A. Likas, "Document clustering using synthetic cluster prototypes," *Data & Knowledge Engineering*, vol. 70, pp. 284–306, 2011.

[13] D. Modha and S. Spangler, "Feature weighting in k-means clustering," *Machine Learning*, vol. 52, pp. 217–237, 2003.

[14] D. J. Ketchen and C. L. Shook, "The application of cluster analysis in strategic management research: an analysis and critique," *Strategic management journal*, vol. 17, no. 6, pp. 441–458, 1996.

[15] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, pp. 411–423, 2001.

[16] K. Lang, "20 newsgroups data set." [Online]. Available: http://www.ai.mit.edu/people/jrennie/20Newsgroups/

[17] A. Strehl and J. Ghosh, "Cluster ensembles a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, 2003.

[18] H. Schmid, "Probalistic part-of-speech tagging using decision tree," in *the Int. Conf. on New Methods in Language*, 1994.

[19] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, pp. 846–850, 1971.