Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 00-048

Bisecting K-means and PDDP: A Comparative Analysis

Sergio M. Savaresi and Daniel Boley

September 28, 2000

Bisecting K−means and PDDP: a comparative analysis

Sergio M. Savaresi[?] − Daniel L. Boley[?]

[?] ***Corresponding Author***. *Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza L. da Vinci, 32, 20133, Milan, ITALY. Phone: +39.02.2399.3545. Fax: +39.02.2399.3412. E−mail: savaresi@elet.polimi.it.*

[?] *Department of Computer Science and Engineering, University of Minnesota, 4−192 EE/CSci, 200 Union St SE, Minneapolis, MN 55455, USA. E−mail:boley@cs.umn.edu.*

Abstract. This paper deals with the problem of clustering a data−set. In particular, the bisecting divisive partitioning approach is here considered. We focus on two algorithms: the celebrated K−means algorithm, and the recently proposed Principal Direction Divisive Partitioning (PDDP) algorithm. A comparison of the two algorithms is given, under the assumption that the data set is uniformly distributed within an ellipsoid. In particular, the dynamic behavior of the K−means iterative procedure is studied; for the 2−dimensional case a closed−form model is given.

Keywords. Unsupervised clustering; K−means; Principal Direction Divisive Partitioning; nonlinear dynamic systems.

1. Introduction and problem statement

The problem this paper focuses on is the *unsupervised clustering* of a data−set. The data−set is given by the matrix $M = [x_1, x_2, ..., x_N] \in \Re^{p \times N}$, where each column of *M*, $x_i \in \Re^p$, is a single data−point. This is one of the more basic and common problems in fields like pattern analysis, data mining, document retrieval, image segmentation, decision making, etc. ([GJJ96], [JMF99]).

The specific problem we want to solve herein is the partition of *M* into **two** sub−matrices (or sub−clusters) $M_L \in \Re^{p \times N_L}$ and $M_R \in \Re^{p \times N_R}$, $N_L + N_R = N$. This problem is known as *bisecting divisive* clustering.

Note that by recursively using a divisive bisecting clustering procedure, the data−set can be partitioned into any given number of clusters. Interestingly enough, the clusters so−obtained are structured as a *hierarchical binary*

tree (or a *binary taxonomy*). This is the reason why the bisecting divisive approach is very attractive in many applications (e.g. in document−retrieval/indexing problems − see e.g. [SKV00] and references cited therein).

Among the divisive clustering algorithms which have been proposed in the literature in the last two decades ([JMF99]), in this paper we will focus on two techniques:

- the *bisecting K−means* algorithm;
- the *Principal Direction Divisive Partitioning* (*PDDP*) algorithm.

K−means is probably the most celebrated and widely used clustering technique; hence it is the best representative of the class of iterative centroid−based divisive algorithms. On the other hand, PDDP is a recently proposed technique ([B97], [B98], [BG+00a], [BG+00b]). It is representative of the non−iterative techniques based upon the Singular Value Decomposition (SVD) of a matrix built from the data−set.

The objective of this paper is twofold:

- compare the clustering performance of bisecting K−means and PDDP;
- analyze the dynamic behavior of the K−means iterative algorithm..

In the existing literature, both these issues have been considered only empirically. The performance of PDDP and K−means have been recently studied, and have been reported to be somehow similar, on the basis of a few application examples ([B97], [B98], [BG+00a], [BG+00b]). As for K−means behavior, the main theoretical result known so far is [SI84], where it is shown that the K−means iterative procedure is guaranteed to converge; however, nothing is said about "where" and "how" it converges.

The main contribution of this work is to provide a simple mathematical explanation of some features of K−means and PDDP. This is done under the restrictive assumption that the data are uniformly distributed within a p−dimensional ellipsoid (some comments on this assumption will be given in the next Section). The main results here obtained can be summarized as follows:

- when the number of data−points tends to infinity, bisecting K−means and PDDP converge to the same solution;
- when the number of data−points tends to infinity, the iterative bisecting K−means algorithm is characterized by p stationary−points; among them, p−1 are unstable equilibria; the remaining one is a stable equilibrium point;
- when the number of data is finite (and small), K−means is strongly initialization−dependent, since it may be trapped into local minima. PDDP guarantees a unique solution but the PDDP solution might be slightly sub−optimal with respect to the best K−means solution. The best use of K−means and PDDP is to use PDDP for the initialization of K−means.

The paper is organized as follows: in Section 2 K−means and PDDP are concisely recalled and discussed; in Section 3 they are analyzed when the number of data−points tends to infinity, whereas in Section 4 an empirical

analysis in the case of finite data sets is proposed. Some concluding remarks end the paper.

2. Bisecting K−means and PDDP

As already stated in the Introduction, this paper focuses on two bisecting divisive partitioning algorithms, which belong to different classes of methods: K−means is the most popular iterative centroid−based divisive algorithm; PDDP is the latest development of SVD−based partitioning techniques. The specific algorithms considered herein are now recalled and briefly commented. In such algorithms the definition of "centroid" will be used extensively; specifically, the centroid of *M*, say w, is given by

$$w = \frac{1}{N}\sum_{j=1}^{N} M_j \,, \tag{1}$$

where M_j is the *j*−th columns of M. Similarly, the centroids of the sub−clusters M_L and M_R, say w_L and w_R, are given by:

$$\begin{cases} w_L = \dfrac{1}{N_L}\displaystyle\sum_{j=1}^{N_L} M_{L,j} \\ w_R = \dfrac{1}{N_R}\displaystyle\sum_{j=1}^{N_R} M_{R,j} \end{cases} \tag{2}$$

where $M_{L,j}$ and $M_{R,j}$ are the *j*−th columns of M_L and M_R, respectively.

Bisecting K−means: the algorithm.

Step 1. (Initialization). Randomly select a point, say $c_L \in \Re^p$; then compute the centroid *w* of *M* (see (1)), and compute $c_R \in \Re^p$ as $c_R = w - (c_L - w)$.

Step 2. Divide $M = [x_1, x_2, ..., x_N]$ into two sub−clusters M_L and M_R, according to the following rule:

$$\begin{cases} x_i \in M_L \quad if \quad \|x_i - c_L\| \le \|x_i - c_R\| \\ x_i \in M_R \quad if \quad \|x_i - c_L\| > \|x_i - c_R\| \end{cases}$$

Step 3. Compute the centroids of M_L and M_R, w_L and w_R, as in (2).

Step 4. If $w_L = c_L$ and $w_R = c_R$, stop. Otherwise, let $c_L := w_L$, $c_R := w_R$ and go back to Step 2.

Bisecting K−means: remarks.

The algorithm above presented is the bisecting version of the general K−means algorithm (as such, it could be properly named "2−means" algorithm?). This bisecting algorithm has been recently discussed and emphasized in [SKV00] and [WW+97]. It these works it is claimed to be very effective in document−processing problems. It is here worth noting that the algorithm above recalled is the very classical and basic version of K−means, also known (see [F65], [GJJ96]) as *Forgy's algorithm* (with a slight modification of the initialization step). Many variations of this basic version of the algorithm have been proposed, aiming to reduce the computational demand, at the price of (hopefully little) sub−optimality. Since the goal of this paper is to analyze convergence properties and clustering performance, this original version of the K−means algorithm is the most interesting and meaningful. ⑦

PDDP: the algorithm.

Step 1. Compute the centroid *w* of M as in (1).

Step 2. Compute the auxiliary matrix $\tilde{M}$ as:

$$\tilde{M} = M - we,$$

where *e* is a *N*−dimensional row vector of ones, namely $e = [1,1,1,1,1,...1]$.

Step 3. Compute the Singular Value Decompositions (SVD) of $\tilde{M}$:

$$\tilde{M} = U\Sigma V^T,$$

where Σ is a diagonal $p \times N$ matrix, and *U* and *V* are ortonormal unitary square matrices having dimension $p \times p$ and $N \times N$, respectively (see [GV96] for an exhaustive description of SVD).

Step 4. Take the first column vector of *U*, say $u = U_1$, and divide $M = [x_1, x_2, ..., x_N]$ into two sub−clusters M_L and M_R, according to the following rule:

$$\begin{cases} x_i \in M_L & if \quad u^T(x_i - w) \le 0 \\ x_i \in M_R & if \quad u^T(x_i - w) > 0 \end{cases}.$$

PDDP: remarks.

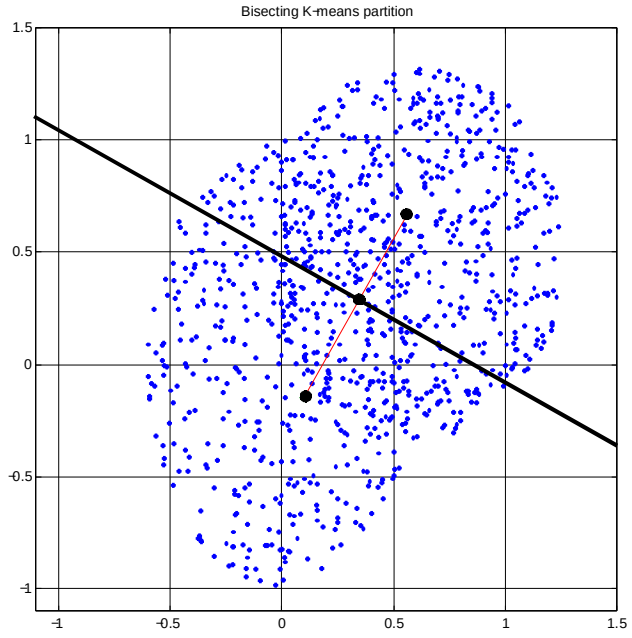
The PDDP algorithm, recently proposed in [B98], belongs to the class of SVD−based data−processing algorithms ([BDO95], [BDJ99]); among them, the most popular and widely known are the *Latent Semantic Indexing* algorithm (LSI − see [A54], [DD+90]), and the LSI−related *Linear Least Square Fit* (LLSF) algorithm ([CY95]). PDDP and LSI mainly differ in the fact that the PDDP splits the matrix with hyperplane passing

through its centroid; LSI through the origin. Another major feature of PDDP is that the SVD of $\tilde{M}$ (Step 3.) can be stopped at the first singular value/vector. This makes PDDP significantly less computationally demanding than LSI, especially if the data−matrix is sparse and the principal singular vector is calculated by resorting to the Lanczos technique ([GV96], [L50]).⑦

The main difference between K−means and PDDP is that K−means is based upon an **iterative** procedure, which, in general, provides different results for different initializations, whereas PDDP is a "**one−shot**" algorithm, which provides a unique solution. In order to understand better how K−means and PDDP work, in Fig.1a and Fig.1b the partition of a generic matrix of dimension 2×2000 provided by K−means and PDDP, respectively, is displayed. From Fig.1, it is easy to see how K−means and PDDP work:

- the bisecting K−means algorithm splits *M* with hyperplane which passes through the centroid *w* of *M*, and is perpendicular to the line passing through the centroids w_L and w_R of the sub−clusters M_L and M_R. This is due to the fact that the stopping condition for K−means iterations is that each element of a cluster must be closer to the centroid of that cluster than the centroid of any other cluster.
- PDDP splits *M* with an hyperplane which passes through the centroid *w* of *M*, and is perpendicular to the principal direction of the "unbiased" matrix $\tilde{M}$ (note that $\tilde{M}$ is the translated version of *M*, having the origin as centroid). The principal direction of $\tilde{M}$ is its direction of maximum variance (see [GV96]).

At a first glance, the two clusters provided by K−means and PDDP look almost indistinguishable. A more careful analysis reveals that the two partitions differ by a few points. Note that this is somewhat unexpected, since the two algorithms differ substantially.



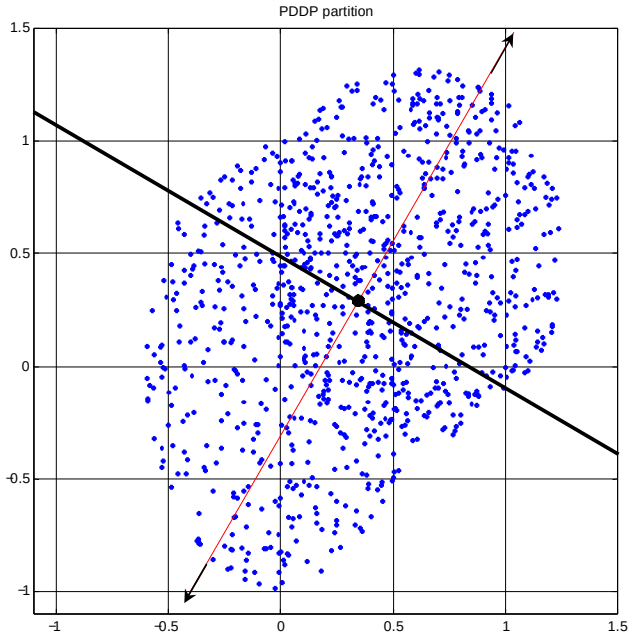


Fig.1a. Partitioning line (bold) of bisecting K−means algorithm. The bullets are the centroids of the data−set and of the two sub−clusters.

Fig.1b. Partitioning line (bold) of PDDP algorithm. The bullet is the centroid of the data set. The two arrows show the principal direction of $\tilde{M}$.

In the rest of the paper we will try to give a rational explanation to the fact that PDDP and bisecting K−means may provide similar results. This will be done by analyzing the dynamic behavior of K−means iteration. Moreover, we will try to clearly outline the *pros* and *cons* of these two seemingly equivalent algorithms.

The analysis presented in the following two sections is based upon the restrictive assumption that the points of the data−set are uniformly distributed within an ellipsoid. This assumption deserves a few words of comment:

- It is important pointing out that an answer to the question "**where** does K−means **converge**?" can be found only if an assumption of the data−distribution is made. Note that this is not mandatory if one only wants an answer to the question "**does** the K−means iteration **converge**?" (as a matter of fact in [SI84] no assumptions on the data distribution are made). Therefore, the sensible choice of the data distribution becomes the main issue.
- Ellipsoid−shaped uniform distribution is the simplest distribution with compact support that, from the clustering point of view, is equivalent to multi−dimensional Gaussian distribution (which is the most typical distribution of experimental data). Henceforth it can be considered the "default" distribution when no a−priori information on the data is available.

3. Theoretical results for infinite data sets

In this section the "asymptotic" behavior of bisecting K−means and PDDP will be analyzed. Asymptotic here means that the data set has an infinite number of points, namely $N \to \infty$. In Fig.2 the difference between a finite and an infinite set of points is naively depicted.

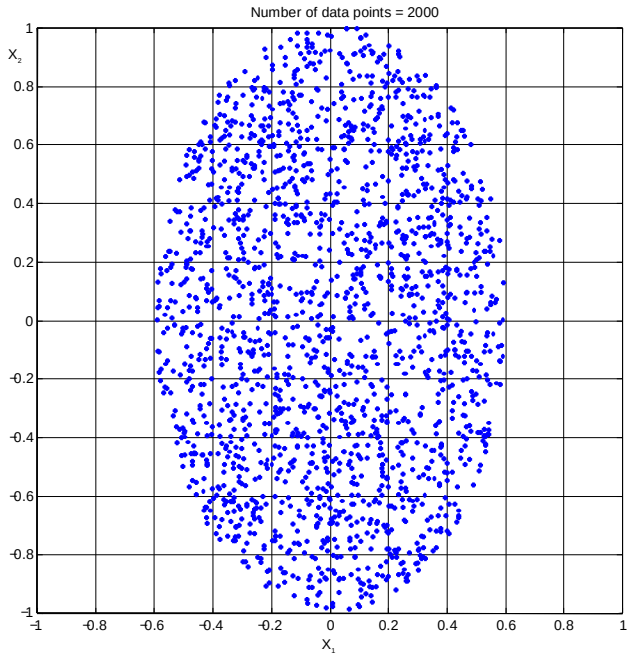


Fig.2a. 2000 data points uniformly distributed within an ellipsoid.

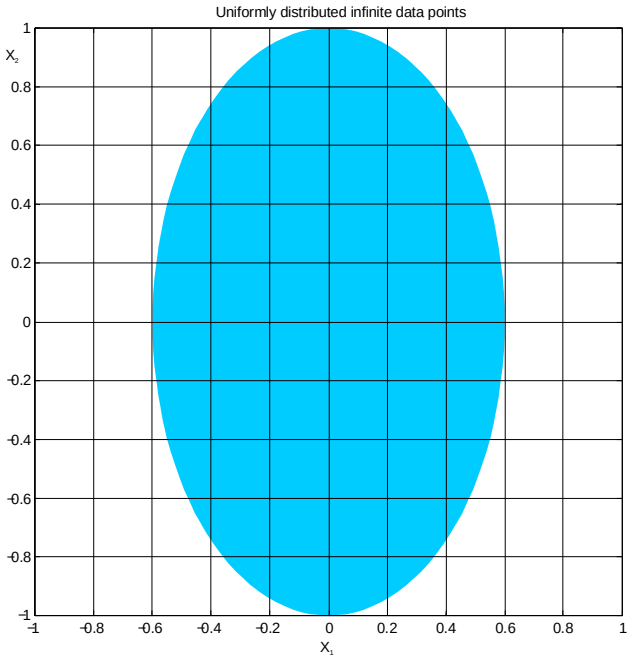


Fig.2b. Infinite data points uniformly distributed within an ellipsoid.

In the first part of this Section, we will focus on the *2−dimensional case*; specifically, it is assumed that each point $x=[x_1,x_2]^T$ of the data−set belongs to an ellipsoid centered in the origin and referred to the axes:

$x=[x_1,x_2]^T$ belongs to the data set if: $\frac{x_1^2}{a^2}+x_2^2\le 1$. (3)

The semi−axes lengths of the ellipsoid in (3) are *a* ($0<a\le 1$) and 1, respectively.

Given these assumptions, the problem we wish now to solve is the mathematical description of the dynamic behavior of the bisecting K−means algorithm. The solution of this problem will be given in the following four items (a)−(d).

(a) Parametrization of the splitting line. First note that the splitting hyperplane (the splitting line in 2−dimensions) is always a line passing through the origin. This property is preserved even at the first step (see the initialization procedure used in Step.1 − Section 2). Henceforth, the splitting line can be parameterized using one parameter only. The natural choice for this parameter is the angle, say α, between the splitting line and the positive x_1 semi−axis. We shall use the subscript "*t*" to indicate the iteration number, namely α_t is the value of α at iteration *t*. With no loss of generality it is also assumed that $0\le\alpha_t\le\pi/2$.

(b) Description of the basic idea. The basic idea used to compute the mathematical model of the dynamic behavior of bisecting K−means is the following. Given α_t, the next angle α_{t+1} can be calculated by first

computing the centroids, say $w_L(\alpha_t)$ and $w_R(\alpha_t)$, of the two semi−clusters induced by the splitting line with angle α_t. The angle α_{t+1} of the next−iteration splitting line then can be easily computed: it is known to be perpendicular to the line connecting $w_L(\alpha_t)$ and $w_R(\alpha_t)$. In this way we obtain a recursive relationship $\alpha_{t+1} = f(\alpha_t)$, which provides a complete description of the dynamic behavior of bisecting K−means.

(c) Computation of the centroids. Due to the infinite number of uniformly distributed points in the data−set, the centroids of the two sub−clusters induced by the splitting line with angle α_t must be computed using integral calculus. Using x_2 as integration variable, the computation of the position of w_L (which is the centroid of the "Left" cluster, bordered with a dashed line in Fig.3) must be split into the computation of the centroids of two sub−pieces of the Left cluster (which are separated by the dashed−dotted line in Fig.3).

The position of w_L hence is given by:

$$w_L = \begin{bmatrix} w_{L1} \\ w_{L2} \end{bmatrix} = \begin{bmatrix} \dfrac{\int_{-S}^{S} \frac{1}{2}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 - a\sqrt{1-x_2^2}\right)\cdot\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2}{\int_{-S}^{S}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2} \\ \dfrac{\int_{S}^{1} x_2 \cdot 2a\sqrt{1-x_2^2}\,dx_2}{\int_{S}^{1} 2a\sqrt{1-x_2^2}\,dx_2} + \dfrac{\int_{-S}^{S} x_2 \cdot \left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2}{\int_{-S}^{S}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2} \end{bmatrix}, \quad (4)$$

where S is the x_2−coordinate of the intersection between the splitting line and the ellipsoid in the first quadrant (see Fig.3); its expression is given by:

$$S = \frac{a \cdot sin(\alpha_t)}{\sqrt{\cos^2(\alpha_t) + a^2 sin^2(\alpha_t)}}. \quad (5)$$

Both (4) and (5) hold for $0 < a \le 1$ and $0 \le \alpha_t \le \pi/2$. Fortunately, (4) can be explicitly computed and significantly simplified. After some cumbersome manipulations it can be shown that w_L is given by:

$$w_L = \begin{bmatrix} w_{L1} \\ w_{L2} \end{bmatrix} = \begin{bmatrix} -\frac{4}{3}\dfrac{a^2 sin(\alpha_t)}{\pi\sqrt{\cos^2(\alpha_t) + a^2 - a^2\cos^2(\alpha_t)}} \\ \frac{4}{3}\dfrac{\cos(\alpha_t)}{\pi\sqrt{\cos^2(\alpha_t) + a^2 - a^2\cos^2(\alpha_t)}} \end{bmatrix}, \; 0 < a \le 1, \; 0 \le \alpha_t \le \pi/2.$$

It is trivial to see that w_R is given by $w_R = -w_L$.

(d) The dynamic model of bisecting K−means. Once $w_L(\alpha_t)$ and $w_R(\alpha_t)$ have been found, it is easy to compute the recursive function $\alpha_{t+1} = f(\alpha_t)$ which models the transition from α_t to the angle α_{t+1} of the next−iteration splitting line. Indeed, this line must be perpendicular to the line passing through w_L and w_R, namely:

$$\alpha_{t+1} = \operatorname{atan}\left[a^2 \tan(\alpha_t)\right],\ 0 < a \le 1,\ 0 \le \alpha_t \le \pi/2. \qquad (6)$$

Equation (6) is one of the major results of this work, since it provides a rigorous closed−form explicit expression of the dynamic behavior of bisecting K−means in the limiting case. Note that (6) represents a first order autonomous (i.e. without forcing inputs) non−linear dynamic discrete−time system. As such, it can be analyzed using non−linear systems theory (see e.g. [L86], [V93]). The analysis of (6) reveals that:

- By solving the steady−state equation

  $$\bar{\alpha} = \operatorname{atan}\left[a^2 \tan(\bar{\alpha})\right],$$

  it is easy to see that the iterative K−means procedure can only have two stationary−points, at $\bar{\alpha} = 0$ and $\bar{\alpha} = \pi/2$. In correspondence to these points the ellipsoid is divided by its shorter axis ($\bar{\alpha} = 0$), and by its longer axis ($\bar{\alpha} = \pi/2$), respectively.

- By locally linearizing the dynamic system (6) about the admissible equilibrium points (namely by computing the tangent model $\delta\alpha_{t+1} = \left(\partial f(\alpha_t)/\partial\alpha_t\right)_{\alpha_t=\bar{\alpha}} \delta\alpha_t$, where $\delta\alpha_t := \alpha_t - \bar{\alpha}$), we obtain the following two linear dynamic discrete−time systems:

  Local dynamic behavior about $\bar{\alpha} = 0$: $\delta\alpha_{t+1} = (a^2)\delta\alpha_t$, $\delta\alpha_t := \alpha_t - 0$;

  Local dynamic behavior about $\bar{\alpha} = \pi/2$: $\delta\alpha_{t+1} = (1/a^2)\delta\alpha_t$, $\delta\alpha_t := \alpha_t - \pi/2$.

  From linear discrete−time dynamic system theory we know that, if $0 < a < 1$, the linear system about $\bar{\alpha} = 0$ is **asymptotically stable**, and the linear system about $\bar{\alpha} = \pi/2$ is **unstable** (indeed they have poles in a^2 and in $1/a^2$, respectively). This means that bisecting K−means **always converges towards** $\bar{\alpha} = 0$, unless the algorithm is exactly initialized with $\alpha_0 = \pi/2$ (namely the initial point c_L exactly belongs to the x_1 – axis). In Fig.4 the function (6) is displayed, when *a*=0.6, and a simulated movement of system (6) is illustrated. Note that, whatever α_0 is (except in the case $\alpha_0 = \pi/2$) the dynamic system $\alpha_{t+1} = f(\alpha_t)$

always converges in $\overline{\alpha} = 0$.

- The value of *a* strongly affects the number of iterations taken by the algorithm to converge. Thanks to equation (6) this number can be given an approximate but quantitative estimate, using dynamic systems theory. First note that the linear system described by the recursive equation $\delta\alpha_{t+1} = (a^2)\delta\alpha_t$ only asymptotically converges at its equilibrium point. A measure of the "speed" at which the system converges towards the equilibrium is given by the so−called *time−constant* τ. τ is defined as the number of steps that $\delta\alpha_t$ takes to decrease its distance from 0 by a factor $1/e$, and it is related to *a* by the following relationship:

$$\tau = \left(-\frac{1}{\log(a^2)} \right).$$

Due to the discrete nature of the distribution, the bisecting K−means algorithm converges in a finite number of steps, say *T*. *T* is a function of the number of the data−points *N* (namely it depends on how densely the data are distributed), which is expected to be **proportional to** τ, namely:

$$T = \gamma(N) \cdot \left(-\frac{1}{\log(a^2)} \right). \qquad (7)$$

The exact value of $\gamma(N)$ is hard to be predicted exactly. A rule−of−thumb typically used by the control systems practitioner can be used to have an idea of $\gamma(N)$: this rule says that, when $\delta\alpha_t$ has reached the 98% of the distance between the initial value and the equilibrium, the system can be considered, in practice, at steady−state. It is easy to see that this corresponds to $\gamma(N) \approx 4$. In Section 4 a numerical validation of this formula will be provided.

Finally note that *T* may take very different values. For instance (if $\gamma(N) \approx 4$), K−means is expected to take only 10−15 iterations to converge if *a*=0.7, about 40 iterations are needed if *a*=0.9, whereas if *a*=0.95 the algorithm might need 80 iterations to converge. It is important to observe, however, that (7) is expected to provide a reliable estimate of *T* only if the number of the points of the data−set is large. For small data−sets the number of iterations required by K−means can be considerably smaller than (7).

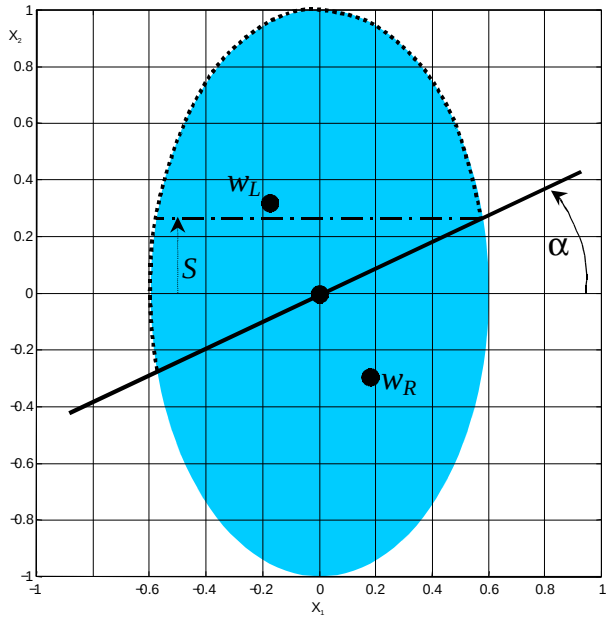


Fig.3. Parametrization of the splitting−line in K−means.

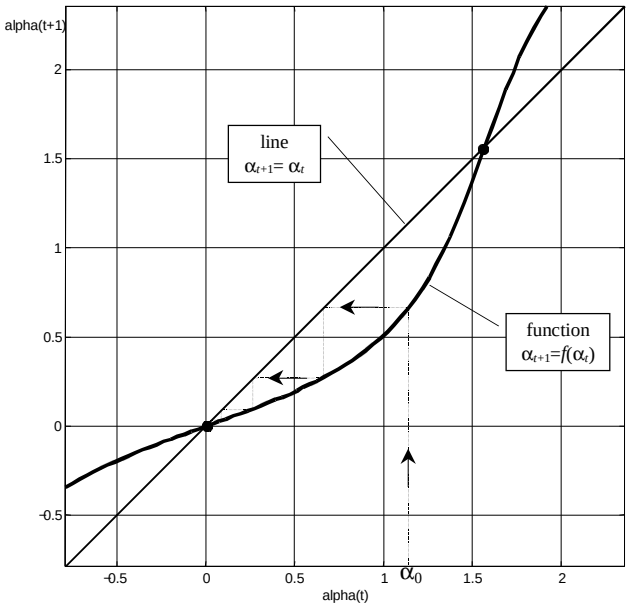


Fig.4. Function (6) (extended over the range $[-\pi/4;3\pi/4]$) when a=0.6. The bullets are the equilibria. The thin line is a simulated movement of (6).

The analysis above presented is the main contribution of this Section. It can be concisely summarized with the following two propositions, generalized to p dimensions.

Proposition 1. If the data−points of a data−set are uniformly distributed in a p−dimensional ellipsoid, the semi−axes of the hyper−ellipsoid have lengths equal to 1, a_1, a_2,?, a_{p-1} ($0<a_i<1$, i=1,2,..,p−1), and $N \to \infty$, then the dynamic discrete−time system which models the K−means iterative algorithm is characterized by p equilibrium points; p−1 points are locally unstable, and 1 is locally stable. In particular, if p=2, the dynamic model has the form: $\alpha_{t+1} = \text{atan}(a^2 \tan(\alpha_t))$, $0 < a < 1$, $0 \le \alpha_t \le \pi/2$. The splitting hyperplanes corresponding to the equilibrium points pass through the origin and are orthogonal to the main axes of the ellipsoid. The splitting hyperplane corresponding to the stable equilibrium point is orthogonal to the largest axis of the ellipsoid.

Proof. When p=2 the proof of this result is given in items (a)−(d) above. Those results can be straightforwardly extended to the p−dimensional case (except for the expression of the dynamic model of the K−means behavior, which, in general, takes a much more complicated expression than (6)).⑦

Proposition 2. If the data−points of a data−set are uniformly distributed in a p−dimensional ellipsoid, the semi−axes of the hyper−ellipsoid have lengths equal to 1, a_1, a_2,?, a_{p-1} ($0<a_i<1$, i=1,2,..,p−1), and $N \to \infty$, then the PDDP algorithm splits the ellipsoid with an hyperplane passing through the origin and orthogonal to the largest axis of the ellipsoid.

Proof. This result is a direct implication of the properties of the SVD. Indeed the p singular vectors of a set of points uniformly distributed within an ellipsoid coincide with the direction of the principal axes of the ellipsoid (see [GV96] for details). ⑦

Propositions 1 and 2 show that bisecting K−means and PDDP provide the same solution, except in the case when the initialization of K−means exactly corresponds to an unstable equilibrium point of the K−means dynamic model. However, if the initialization is made randomly, this event occurs with probability zero.

These asymptotic results are useful to gain a deep insight into the bisecting K−means algorithm, and to explain why, in many cases, K−means and PDDP show a very similar clustering behavior. However, when the data set contains a finite number of data (namely when the number of points is comparatively small), bisecting K−means and PDDP might provide solutions, which, sometimes, are remarkably different. The finite data−set case will be analyzed and discussed in the next section, on the basis of numerical results obtained by simulation.

4. Numerical results for finite data sets

In this section, the bisecting K−means and PDDP will be analyzed when the data−set has a finite number of data−points. The analysis will be done empirically, using simulated data.

The purpose of this section is twofold:

- validate the theoretical results obtained in the previous section, and see how they change when the data−set is finite;
- understand the *pros* and *cons* of K−means and PDDP.

The analysis is structured as follows: in Subsection 4.1 the dynamic model of K−means will be numerically computed for finite data−sets, and the problem of local minima will be discussed; in Subsection 4.2 the formula (7) for the estimation of the number of iterations required by K−means to converge will be validated; in Subsection 4.3 the clustering performance of K−means and PDDP will be compared. Finally, in Subsection 4.4 some conclusions on the *pros* and *cons* of the two algorithms will be drawn.

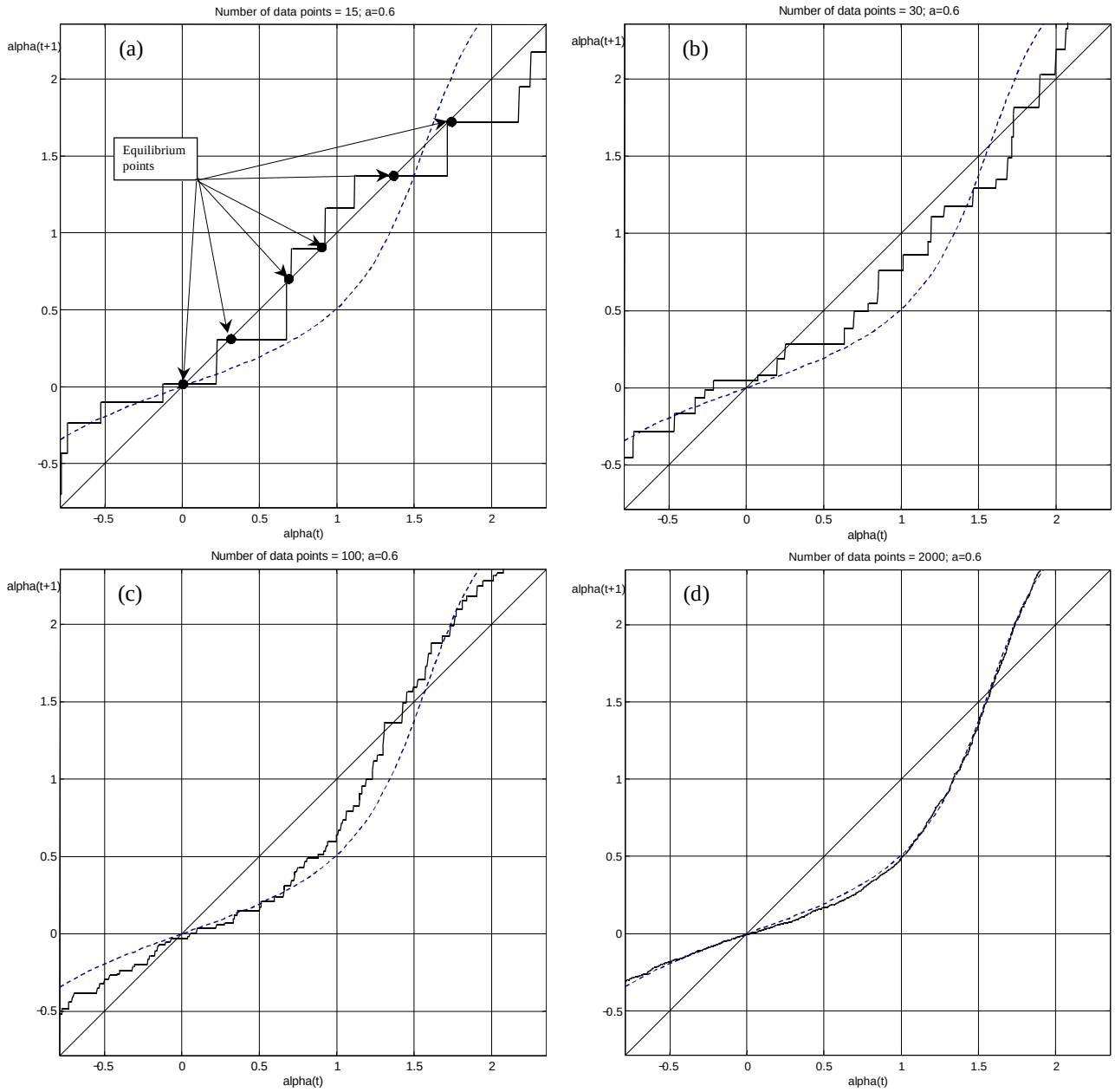


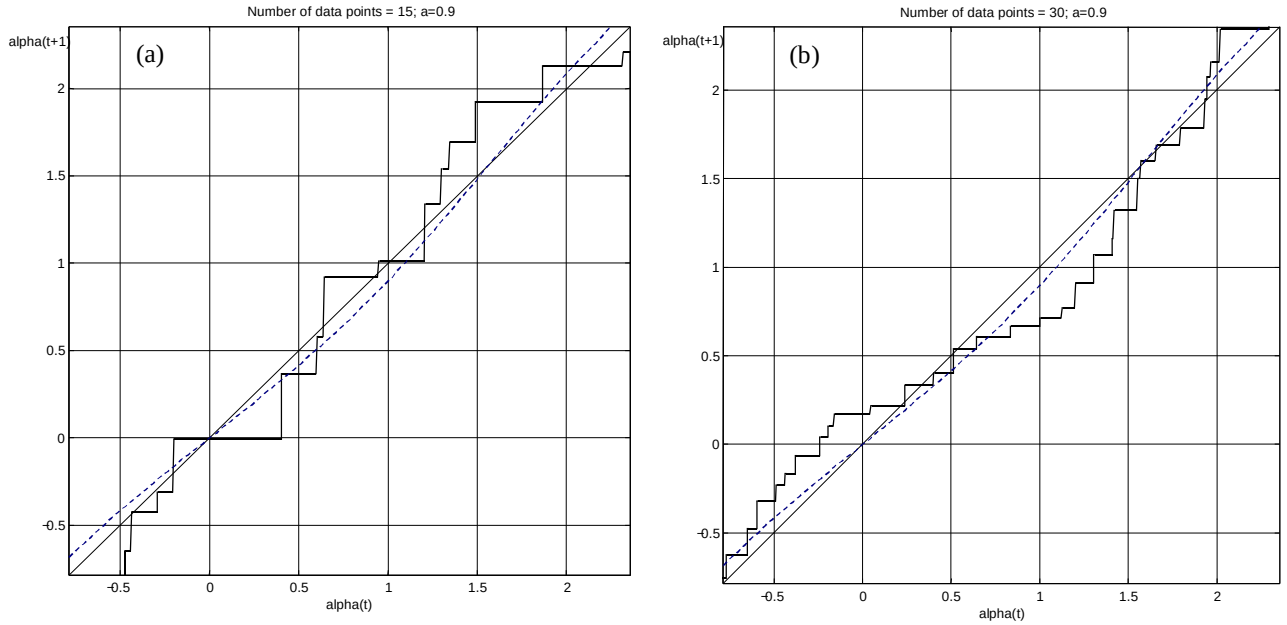
Fig.5. Recursive function $\alpha_{t+1} = f(\alpha_t)$ estimated from data, when ***a*=0.6**. The dashed line is the asymptotic function (6) computed in Section 4. (a): *N*=15; (b): *N*=30; (c): *N*=100; (d): *N*=2000.

4.1. The dynamic model of K−means and the problem of local minima

The first problem we consider is the analysis of the K−means dynamic behavior when the data−set has a finite number of data. As a first experiment, four sets of data have been considered, characterized by 15, 30, 100 and 2000 data−points uniformly distributed within a 2−dimensional ellipsoid with ***a*=0.6**. The recursive function $\alpha_{t+1} = f(\alpha_t)$ has been numerically computed for these four data−sets. The results are displayed in Fig.5. From the inspection of Fig.5, the following remarks can be done:

- The main difference between the asymptotic function (6) and the recursive functions corresponding to finite data−sets is that the latter are step−wise functions. A major consequence of this function being step−like is that every equilibrium point (namely every point where the function crosses the line $\alpha_{t+1} = \alpha_t$ – see Fig.5a) is locally asymptotically stable, since the local slope of the function about the equilibrium is smaller than 1. Note that this explains why, in the case of finite data−sets, the K−means algorithm is affected by "local minima" problems.
- When the number of data−point grows, the finite data−set function converges towards the asymptotic function (see Fig.5d). This validates the theoretical model developed in the previous section. Moreover, notice that when the number of data−point gets large, the number of equilibrium points decreases, and each step gets narrower (see e.g. Fig.5c). This explains why, when the number of data is sufficiently large, it is the common experience that the problem of local minima tends to vanish.

As a second experiment, the recursive function $\alpha_{t+1} = f(\alpha_t)$ has been computed for four sets of 15, 30, 100 and 2000 data−points uniformly distributed within a 2−dimensional ellipsoid with ***a*=0.9**. The results are displayed in Fig.6. The main difference in the results between the case ***a*=0.6** and ***a*=0.9** is that in the latter the problem of multiple equilibrium points is more severe.



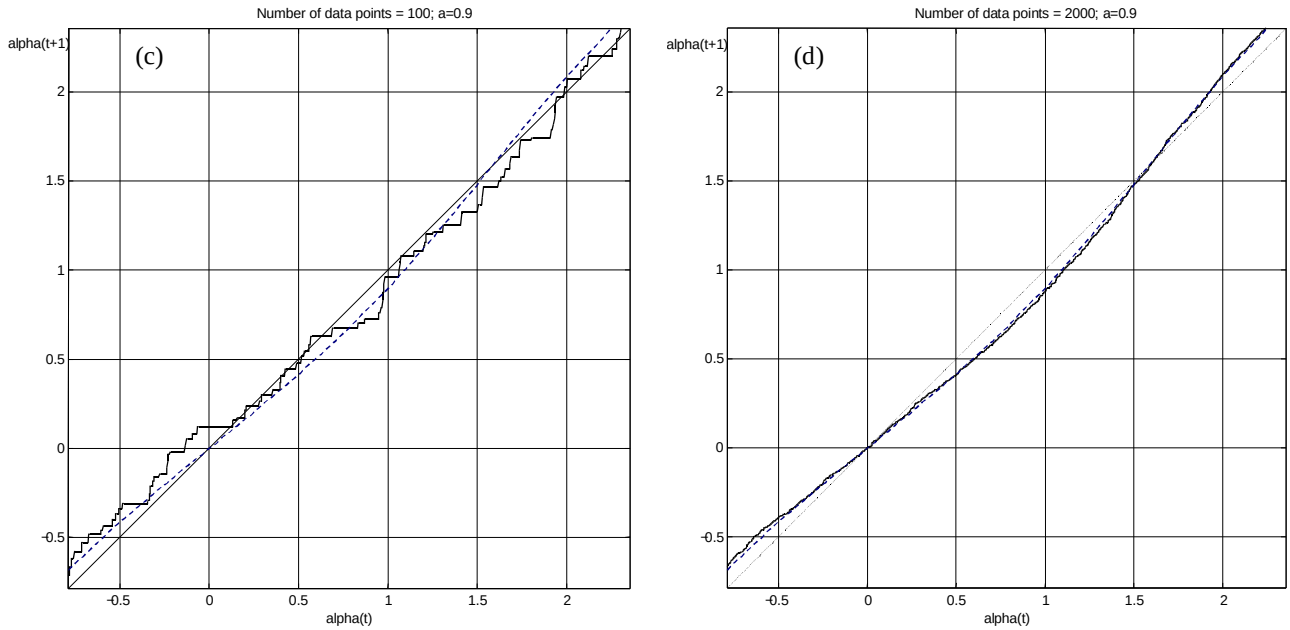


Fig.6. Recursive function $\alpha_{t+1} = f(\alpha_t)$ estimated from data, when ***a*=0.9**. The dashed line is the asymptotic function (6) computed in Section 4. (a): *N*=15; (b): *N*=30; (c): *N*=100; (d): *N*=2000.

The above experiments suggest that the problem of local minima for bisecting K−means is expected to:

- *decrease* when the number of data grows;
- *increase* when the size of the "short" semi−axes ($a_1,...,a_{p-1}$) approaches the largest axis.

In order to validate these conjectures, the bisecting K−means algorithm has been extensively tested for different values of *a* (*a*=0.6,0.7,0.8,0.9) and for different sizes of the data−set (*N* ranging from 10 to 5000). The average dispersion of the centroids we have obtained (which is directly related to the problem of local minima) is displayed in Fig.7. In particular, for each value of *N*, 20 different data−sets have been randomly generated; for each data−set, 100 different runs of K−means have been done (starting from different initial conditions), so obtaining 100 "dispersed" centroids. The dispersion of these 100 centroids has been computed for each of the 20 data−sets, and averaged. Note that the conjectures above outlined are fully confirmed by the data: the centroids dispersion increases with *a*, and decreases with *N*.

4.2. The time of convergence of K−means iterations

An interesting result proposed in Section 4, which must be validated, is the prediction of the number of iterations which bisecting K−means needs to converge. Recall that expression (7) is expected to hold approximately if the data set is very large. For small data−sets the convergence is expected to be faster.

To this end, the number of iterations required by K−means to converge has been experimentally estimated for different values of *a* in the range [0.7,0.95], using data−sets of size *N*=20000. The results are in Fig.8. Notice

the very good fit between the predicted and the estimated results ($\gamma(N)$ used in Fig.8 to predict the number of iterations of K−means is $\gamma(N)=4$, which is the "rule−of−thumb value" suggested in Section 3).

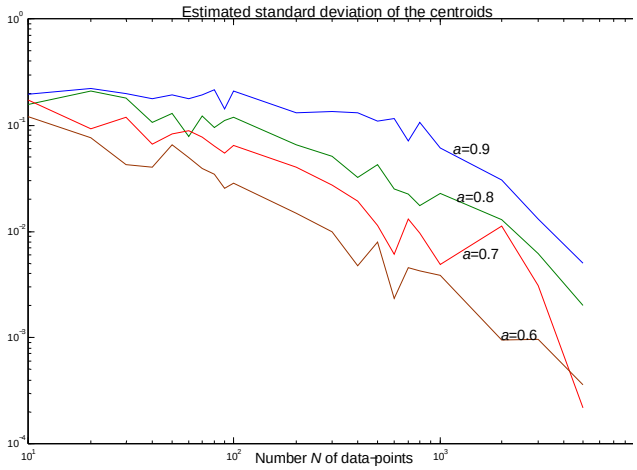


Fig.7. Average dispersion of the centroids M_L and M_R computed via K−means, as a function of the number of data−points. The four lines correspond to different values of *a*.

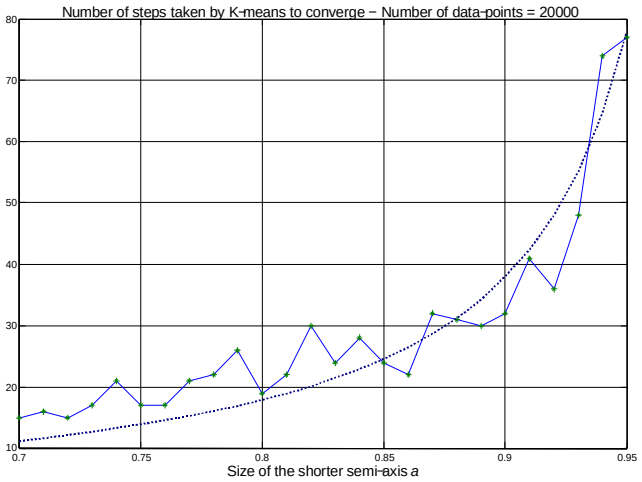


Fig.8. Estimated number of iterations required by K−means to converge, as a function of *a*. The dashed line is the number of iterations predicted by (7), with γ(*N*)=4.

4.3. Comparing the clustering performance of bisecting K−means and PDDP

The last crucial issue we consider is the analysis of the clustering performance of K−means and PDDP. This issue is immaterial when the data−set is very large, since both methods provide the same results, but is very important when the size of the data set is comparatively small.

To make a comparison between the performance of different clustering algorithms, a performance index must be used. Given two sub−matrices $M_L \in \Re^{N_L}$ and $M_R \in \Re^{N_R}$ of the data set $M=[x_1, x_2, ..., x_N] \in \Re^{p\times N}$, a widely−accepted way of measuring the internal quality of the partition is given by the following penalty index (see e.g. [JMF99], [SI84], [SKV00]):

$$J(M_L, M_R) = \sum_{x_i \in M_L} \|x_i - w_L\|^2 + \sum_{x_i \in M_R} \|x_i - w_R\|^2 \quad , \tag{8}$$

where w_L and w_R are the centroids of M_L and M_R, given by (2). Note that (8) is a measure of cohesiveness of each cluster to its centroid: the smaller $J(M_L, M_R)$ is, the better is the partition.

It is worth pointing out that clustering *M* by direct minimization of $J(M_L, M_R)$ would be, conceptually, the best clustering method. Unfortunately, the minimization of (8) is known to require exhaustive search which is exponential in time with respect to the number of data−points. Note that the clustering algorithms which have

been proposed in the literature (including K−means and PDDP) can be interpreted as *alternate ways of tackling the problem of minimizing* (8). All of them provide a solution with a reasonable computational effort, at the price of some sub−optimality.

In order to compare the performance of K−means and PDDP, we have considered two sets of data, uniformly distributed in a 100−dimensional ellipsoid. The main semi−axis of the ellipsoid is 1. The size of the remaining 99 semi−axes is in the range [0.05, 0.95]. The first data−set has 1000 points. The second data−set 5000. Note that this number of data−points is comparatively small for a 100−dimensional vector space.

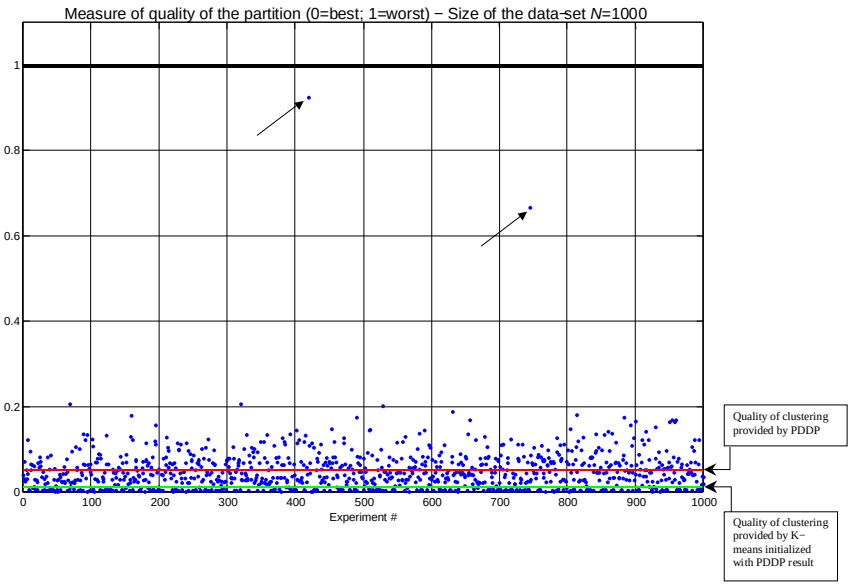


Fig.9. Measure of quality of bisecting partition of a data−set of *N*=1000 points.

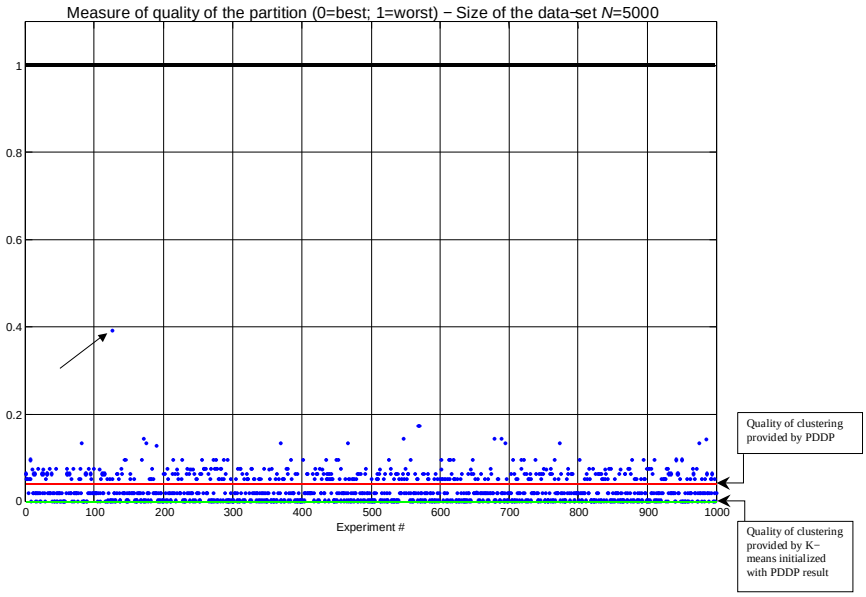


Fig.10. Measure of quality of bisecting partition of a data−set of *N*=5000 points.

For each data set, the following clustering techniques have been used:

(a) Bisecting K−means, initialized randomly. Specifically, 1000 different initializations have been tested for each data−set.

(b) PDDP.

(c) Bisecting K−means, initialized with the result provided by PDDP.

At the end of each clustering experiment, the so−obtained partition has been evaluated using (8). The results are displayed in Fig.9 (N=1000) and in Fig.10 (N=5000). The measure of quality in Figs.9−10 is a normalized version of (8). Specifically, 0 corresponds to the best clustering performance we have found; 1 corresponds to the "worst−case" situation of non−partitioned cluster (namely $M_L = M$ and $M_R = \varnothing$). The 1000 dots show the clustering performance of K−means randomly initialized; the two horizontal lines show the performance of PDDP, and the performance of K−means initialized via PDDP.

From the inspection of Figs.9−10, the following remarks can be done:

- The results obtained by random initialization of K−means suffer a remarkable large variation: the corresponding performance index is spread within the [0,0.2] range. Moreover, notice that K−means may converge towards very "bad" (in terms of clustering performance) solutions (which are indicated with an arrow in Figs.9−10). In Fig.9 there are two "bad" solutions, characterized by a 70% and a 90% (!) performance loss, whereas in Fig.10 the worst solution is characterized by a 40% performance loss.
- In both cases, PDDP slightly under−performs (of about 5%) the best result obtained by K−means. However, it outperforms the worst and the average results of K−means.
- The combination of PDDP and K−means provides very good results. In particular, in the case N=1000 the final performance loss with respect to the best K−means solution is about 1%; in the case N=5000 there is no loss of performance.

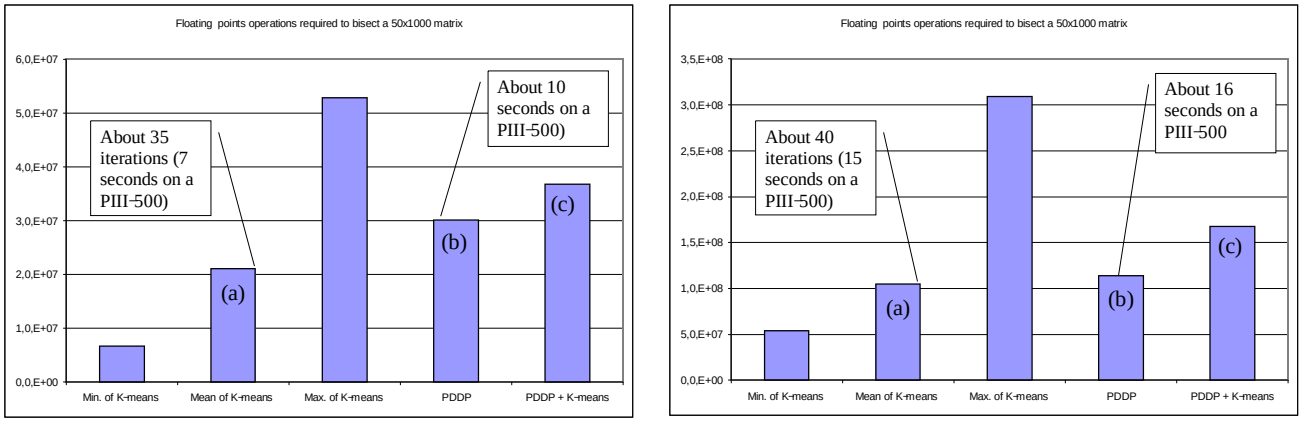


Fig.11. Comparison of the computational effort required by methods (a)−(c).

To complete this analysis, a few words on the computational power required by the clustering experiments (a)−(c) must be said. To this end, the number of floating point operations (*flops*) spent to cluster the 100×1000

and the 100×5000 data matrices are displayed in Fig.11. The following can be said:

- The PDDP requires about the number of *flops* required *in average* by a run of K−means. This means that that PDDP must be compared with the result of a single run of K−means. In light of the performance results displayed in Figs.9−10, at equal computational power PDDP is expected to provide better performance than K−means.
- As expected, the computational effort required by a run of K−means varies a lot: the minimum and the maximum values of *flops* may differ of an order of magnitude.
- The refinement of the PDDP solution with a run of K−means requires little additional computational power (which − as one intuitively expects − is approximately equal to the minimum of *flops* required by K−means randomly initialized). This is due to the fact that PDDP provides a solution which is close to an equilibrium point for K−means.

Obviously, the above quick comparison of K−means and PDDP computational demand is far from being exhaustive: the computational power depends on many variables, and is significantly *implementation−dependent* and *data−dependent* (a complete analysis of this issue goes beyond the scope of the present work). For instance, it can be shown that if M tend to be "square" (namely $p \approx N$), PDDP is significantly more demanding than a single run of K−means, whereas, if $p << N$, PDDP outperforms K−means (this trend can be observed by comparing the two graphs in Fig.11). However, it is interesting to see that the above results are very consistent with the results one intuitively expects.

4.4. K−means *versus* PDDP: concluding remarks

On the basis of the numerical analysis proposed in this Section, we can briefly summarize the *pros* and *cons* of bisecting K−means and PDDP, when the size of the data−set is comparatively small:

- K−means is very simple to implement, and tends to give slightly better results in terms of partition quality. However, it is not deterministic (its results strongly depend on the initialization), and it might take a large number of iterations to converge. Hence, if the "best" result is searched for, it is significantly more demanding than PDDP, in terms of computational power.
- PDDP gives a deterministic result, and cannot be fooled by local minima. However it tends to provide results which are slightly worse than the best K−means results.

The peculiar features of K−means and PDDP above outlined can be summarized in the following "rule−of−thumb" for the practitioner:

- The best performance in terms of quality of clustering are obtained by running K−means a large numbers of times, with different initializations, and picking up the best result. However this requires a large computational effort.

- The quickest and safest way of obtaining a "reasonably good" solution is using PDDP.
- The best compromise between computational effort and cluster quality is to use K−means initialized with the PDDP result. This procedure has the additional advantage of providing a deterministic result.

5. Conclusions

In this paper the problem of clustering a data−set is considered. Two bisecting divisive clustering techniques are considered: the K−means and the PDDP. The similarity and the differences of these two algorithms are outlined by means of a theoretical and an empirical analysis. In particular, the dynamic behavior of the recursive K−means algorithm is studied, and, under some restrictive assumptions, a closed−form model is developed.

Acknowledgements

First author supported by *Consiglio Nazionale delle Ricerche (CNR) short−term−mobility program*. Second author supported by *NSF grant IIS−9811229*. Thanks are also due to Prof. Gene Golub of *Dept. of Computer Science at Stanford*, to Prof. Sergio Bittanti of *Politecnico di Milano*, and to Prof. Giovanna Gazzaniga of *Pavia CNR Institute of Numerical Analysis*.

References

[A54] Anderson, T. (1954). "On estimation of parameters in latent structure analysis". *Psychometrica*, vol.19, pp.1−10.

[BDO95] Berry M.W., S.T. Dumais, G.W. O'Brien (1995). "Using Linear Algebra for intelligent information retrieval". *SIAM Review*, vol.37, pp.573−595.

[BDJ99] Berry, M.W., Z. Drmac, E.R. Jessup (1999). "Matrices, Vector spaces, and Information Retrieval". *SIAM Review*, vol.41, pp.335−362.

[B97] Boley, D.L. (1997). "Principal Direction Divisive Partitioning". *Technical Report TR−97−056*, Dept. of Computer Science, University of Minnesota, Minneapolis.

[B98] Boley, D.L. (1998). "Principal Direction Divisive Partitioning". *Data Mining and Knowledge Discovery*, vol.2, n.4, pp. 325−344.

[BG+00a] Boley, D.L., M. Gini, R. Gross, S. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, J. Moore (2000). "Partitioning−Based Clustering for Web Document Categorization". *Decision Support Systems* (to appear).

[Bg+00b] Boley, D.L., M. Gini, R. Gross, S. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, J. Moore (2000). "Document Categorization and Query Generation on the World Wide Web Using WebACE". *AI Review* (to appear).

[CY95] Chute, C., Y. Yang (1995). "An overview of statistical methods for the classification and retrieval of patient events". *Meth. Inform. Med.*, vol.34, pp.104−110.

[DD+90] Deerwester, S., S. Dumais, G. Furnas, R. Harshman (1990). "Indexing by latent semantic analysis". *J. Amer. Soc. Inform. Sci*, vol.41, pp.41−50.

[F65] Forgy, E. (1965). "Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of Classification". *Biometrics*, pp.768−780.

[GV96] Golub, G.H, C.F. van Loan (1996). *Matrix Computations (3rd edition)*. The Johns Hopkins University Press.

[GJJ96] Gose, E., R. Johnsonbaugh, S. Jost (1996). *Pattern Recognition & Image Analysis*. Prentice−Hall.

[JMF99] Jain, A.K, M.N. Murty, P.J. Flynn (1999). "Data Clustering: a Review". *ACM Computing Surveys*, Vol.31, n.3, pp.264−323.

[L50] Lanczos, C. (1950). "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators". *J. Res. Nat. Bur. Stand*, vol.45, pp.255−282.

[L86] LaSalle, J.P. (1986). *The Stability and Control of Discrete Processes*. Springer−Verlag.

[SI84] Selim, S.Z., M.A. Ismail (1984). "K−means−type algorithms: a generalized convergence theorem and characterization of local optimality". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.6, n.1, pp.81−86.

[SKV00] Steinbach, M., G. Karipis, V. Kumar (2000). "A comparison of Document Clustering Techniques". *Proceedings of World Text Mining Conference, KDD2000*, Boston.

[V93] Vidyasagar, M. (1993). *Nonlinear Systems Analysis*. Prentice−Hall

[WW+97] Wang, J.Z., G. Wiederhold, O. Firschein, S.X. Wei (1997). "Content−based image indexing and searching using Daubechies' wavelets". *Int. J. Digit. Library*, vol.1, pp.311−328.

$$w_L = \begin{bmatrix} w_{L1} \\ w_{L2} \end{bmatrix} = \begin{bmatrix} \dfrac{\int_S^1 0 \cdot 2a\sqrt{1-x_2^2}\,dx_2}{\int_S^1 2a\sqrt{1-x_2^2}\,dx_2} + \dfrac{\int_{-S}^{S} \frac{1}{2}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 - a\sqrt{1-x_2^2}\right)\cdot\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2}{\int_{-S}^{S}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2} \\ \dfrac{\int_S^1 x_2 \cdot 2a\sqrt{1-x_2^2}\,dx_2}{\int_S^1 2a\sqrt{1-x_2^2}\,dx_2} + \dfrac{\int_{-S}^{S} x_2 \cdot \left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2}{\int_{-S}^{S}\left(\frac{\cos(\alpha_t)}{\sin(\alpha_t)}x_2 + a\sqrt{1-x_2^2}\right)dx_2} \end{bmatrix}$$