Improving Bregman k-means

Wesam Ashour*

Islamic University of Gaza,
Gaza, Palestine
E-mail: Wesam.Ashour@iugaza.edu.ps
*Corresponding author

Colin Fyfe

University of the West of Scotland,
High Street, Paisley PA1 2BE, Scotland
E-mail: Colin.Fyfe@uws.ac.uk

Abstract: We review Bregman divergences and use them in clustering algorithms which we have previously developed to overcome one of the difficulties of the standard k-means algorithm which is its sensitivity to initial conditions which leads to finding sub-optimal local minima. We show empirical results on artificial and real datasets.

Keywords: clustering; K-means; local optima; Bregman divergences.

Reference to this paper should be made as follows: Ashour, W. and Fyfe, C. (2014) 'Improving Bregman k-means', *Int. J. Data Mining, Modelling and Management*, Vol. 6, No. 1, pp.65–82.

Biographical notes: Wesam Ashour is an Assistant Professor at Islamic University of Gaza. He is an active researcher at the Applied Computational Intelligence Research Unit in the University of the West of Scotland. He obtained his Master and Doctorate degrees from UK. His research interests include data mining, artificial intelligence, reinforcement learning and neural networks.

Colin Fyfe is a Personal Professor at the University of the West of Scotland. He has published more than 300 refereed papers and been Director of Studies for 21 PhDs. He is on the editorial boards of six international journals and has been Visiting Professor at universities in Hong Kong, China, Australia, Spain, South Korea and USA.

1 Introduction

Data clustering is an unsupervised classification method aims at creating groups of objects, or clusters, in such a way that objects in the same cluster are very similar and objects in different clusters are quite distinct. One difficulty in this process is that we do not have any prior knowledge about the structure of the data, or its labels. The k-means algorithm (Hartigan and Wang, 1979; Lloyd, 1982; MacQueen, 1967) is one of the most frequently used investigatory algorithms in data analysis. It starts with initial k cluster

prototypes (centroids), and then it assigns each data point to the nearest prototype, updates the cluster prototypes, and repeats the process until the k prototypes do not change. Although there is no guarantee of achieving global minima, at least the convergence of the k-means algorithm is ensured. Therefore, how to choose proper initial cluster prototypes becomes very important for the k-means algorithm. The greedy essence makes k-means algorithm not converge to the global optimum and its performance strongly depends on the initial guess of partitions. The k-means algorithm gives better results only when the initial partitions are close to the final solution. The impact of initialising prototypes is significant in k-means, so there are several methods have been proposed to solve the cluster initialisation problem. Arai and Barakbah (2007) proposed a hierarchical method to determine the initialisation of clusters by applying k-means several times, then obtaining a set of prototypes from each different run; these prototypes will be treated as a dataset, and handled by a hierarchal clustering algorithm to obtain the best prototypes. Lu et al. (2008) proposed another hierarchical initialisation method to the k-means clustering problem. The core of this method is to treat the clustering problem as a weighted clustering problem so as to find better initial cluster centres based on the hierarchical approach. It depends on two major steps: first it reduces the data from the bottom up, by sampling the clusters; it then maintains clusters at the level that sampling stops, which allows them to obtain clusters prototypes; then based on those prototypes it finds the cluster prototypes of the original data by using a hierarchal method. The algorithm seems to give an acceptable result for low dimensional datasets; it suffers from the curse of the dimensionality with high dimensional datasets. Arthur and Vassilvitskii (2007) improved the k-means algorithm by substituting the random allocation of the prototypes with a seeding technique. They give experimental results that show the advantage of this algorithm in time and accuracy.

Khan and Ahmad (2004) proposed an algorithm for initialising k-means prototypes based on individual attributes of the pattern, which may provide some information about initial cluster centres. The algorithm's main concept is applying k-means for each attribute to compute cluster prototypes for individual attributes. This is done by assuming each of attributes of the pattern space is normally distributed; they then divide the normal curve into k partitions, and apply the k-means algorithm on this attribute. They then allocate previous cluster labels to every pattern, and run k-means on the complete dataset. Now each pattern has a set of class labels; a centre of these classes must be found and used as prototypes for k-means.

Cao et al. (2009) proposed a method for initialising the k-means algorithm using a neighbourhood model. The cohesion degree of the neighbourhood of an object and the coupling degree between neighbourhoods of objects are defined based on the neighbourhood-based rough set model. A new initialisation method is proposed by computing cohesion for each object, and finding the one that has maximum cohesion; this will be considered as the first prototype. The next prototype will be the most coherent object satisfying maximum cohesion (after removing the selected centres). This procedure will be repeated until it finds the required number of prototypes, then for each centre, it must have coupling with the samples below a threshold. If not, it will be removed and the algorithm will try to find the next centre until it finds the desired number of prototypes.

Bregman divergences (Frigyik et al., 2008; Neilsen et al., 2007a, 2007b) have been used for exploratory data analysis to extend existing methods in a number of ways; for example, for projection methods, Collins et al. (2001) gave a Bregmanised version of

principal component analysis, Wang and Fyfe (2010) repeated this for independent component analysis and Wang et al. (2011) did the same for canonical correlation analysis. A very influential paper in this area discussed a Bregmanised version of k-means (Banerjee et al., 2005) in both hard and soft forms, however we have found that this version also suffers from the well-known problems of standard k-means – a sensitivity to initial conditions and hence convergence to a local optimum.

Previously developed algorithms which overcome the problem of convergence to local optima have been discussed in Barbakh and Fyfe (2008a, 2008b) and in this paper, we apply the same techniques to Bregmanised k-means and show that they improve upon the convergence of Bregmanised k-means.

2 Bregman divergences

The Bregman divergence between p and q based on the convex function $F(x)$ is:

$$d_F(p, q) = F(p) - F(q) - (p - q)\nabla F(q) \tag{1}$$

For example, for the convex function $F(x) = 2x \ln x$, then the right divergence between a data sample $\mathbf{x}$ and a prototype/centre $\mathbf{m}$ is given by:

$$\begin{aligned} d_F(\mathbf{x}, \mathbf{m}) &= 2\mathbf{x} \ln \mathbf{x} - 2\mathbf{m} \ln \mathbf{m} - (\mathbf{x} - \mathbf{m})(2 \ln \mathbf{m} + 2) \\ &= 2[\mathbf{x} \ln \mathbf{x} - \mathbf{x} \ln \mathbf{m} - \mathbf{x} + \mathbf{m}] \end{aligned} \tag{2}$$

We call this the right divergence since the trainable prototype is in the right position in the divergence. Similarly, we can consider the left divergence which is given by:

$$\begin{aligned} d_F(\mathbf{m}, \mathbf{x}) &= 2\mathbf{m} \ln \mathbf{m} - 2\mathbf{x} \ln \mathbf{x} - (\mathbf{m} - \mathbf{x})(2 \ln \mathbf{x} + 2) \\ &= 2[\mathbf{m} \ln \mathbf{m} - \mathbf{m} \ln \mathbf{x} - \mathbf{m} + \mathbf{x}] \end{aligned} \tag{3}$$

We can think of Bregman divergences as distance measures while recognising that they are not true distances since they are not symmetric. Examples of Bregman divergences include:

Example 1: The squared Euclidean distance is a special case of the Bregman divergence in which $F(.) = \| . \|^2$

$$\begin{aligned} d_F(\mathbf{x}, \mathbf{y}) &= \| \mathbf{x} \|^2 - \| \mathbf{y} \|^2 - \langle \mathbf{x} - \mathbf{y}, \nabla F(y) \rangle \\ &= \| \mathbf{x} \|^2 - \| \mathbf{y} \|^2 - \langle \mathbf{x} - \mathbf{y}, 2\mathbf{y} \rangle \\ &= \| \mathbf{x} - \mathbf{y} \|^2 \end{aligned}$$

Example 2: The Kullback-Leibler divergence (Dhillon et al., 2003) is another special case in which $F(\mathbf{p}) = \sum_{j=1}^{d} p_j \log p_j$. Consider two discrete probability distributions, $\mathbf{p}$ and $\mathbf{q}$.

$$
\begin{aligned}
d_F(\mathbf{p},\mathbf{q}) &= \sum_{j=1}^{d} p_j \log_2 p_j - \sum_{j=1}^{d} q_j \log_2 q_j - \langle \mathbf{p}-\mathbf{q}, \nabla F(\mathbf{q}) \rangle \\
&= \sum_{j=1}^{d} p_j \log_2 p_j - \sum_{j=1}^{d} q_j \log_2 q_j - \sum_{j=1}^{d} \left(p_j - q_j\right)\left(\log_2 q_j + \log_2 e\right) \\
&= \sum_{j=1}^{d} p_j \log_2 \frac{p_j}{q_j} - \log_2 e \sum_{j=1}^{d} \left(p_j - q_j\right) \\
&= \sum_{j=1}^{d} p_j \log_2 \frac{p_j}{q_j} = K.L.(\mathbf{p} \,||\, \mathbf{q})
\end{aligned}
$$

since $\sum_{j=1}^{d} p_j = \sum_{j=1}^{d} q_j = 1.$

It is straightforward to show (Neilsen and Nock, 2007) that right centroids are unique and independent of which specific Bregman divergence is used; all give the centre of mass, $\mathbf{m} = \frac{1}{N}\mathbf{x}$ although when used in a k-means algorithm, the local minimum reached may be different.

However, the left centroids are different from one another. Finally, Banerjee et al. (2005) show that there is a bijection between the Bregman divergences and members of the exponential distributions of probability density functions. Thus, if we know the distribution of the dataset, we could find the optimal divergence to use with that dataset. In practice, of course, we are rarely in a position to know the data distribution exactly and this must be approximated from samples from a dataset.

3 New Bregman clustering algorithms

The central idea in the new algorithms is that each prototype (sometimes also called the seed or centroid), before being moved to any new location, responds to all the other prototypes' positions and, in particular, to their relative locations with respect to the data points, and hence it is possible to identify the free clusters that are not recognised by the other prototypes. For example if we have two data points (each data point represents a cluster) and two prototypes, one of them being closer to the first data point, the other prototype will, before responding, recognise that there is one prototype closer to the first data point and hence will prefer to move toward the second point.

3.1 Bregman inverse weighted k-means algorithm

Consider the performance function:

$$
J_1 = \sum_{i=1}^{N} \left[\sum_{j=1}^{K} \frac{1}{\left\| \mathbf{x}_i - \mathbf{m}_j \right\|^p} \right] \min_{k=1}^{K} \left\| \mathbf{x}_i - \mathbf{m}_k \right\|^n \tag{4}
$$

where K is the number of clusters, N is the number of data points, and n and p are any power values. This performance function consists of two functions multiplied together,

the minimum function which is used as a performance function for k-means and the inverse weighted function. The main problem of sensitivity to the initial conditions in k-means is generated due to the effect of the minimum function. This minimum function helps to find the clusters, but in the learning process, each prototype is updated based only on its closest data samples. Each prototype has not been affected simultaneously by the data points that have closer prototypes. This may lead to different results when we have different initialisation. In addition, the minimum function causes the possibility to converge to a local optimum. The purpose of the inverse weighted function is to provide a good learning process for all prototypes, without losing the advantage of the minimum function which helps to find the clusters. We need each prototype to respond to all data in the learning process, not only to its members, i.e., those samples which are closest to it. The inverse weighted function provides this target. It provides a relation between all data samples and all prototypes.

Let $\mathbf{m}_{k*}$ be the closest prototype to $\mathbf{x}_i$. Then

$$\begin{aligned} J_1(\mathbf{x}_i) &= \left[\sum_{j=1}^{K} \frac{1}{\left\|\mathbf{x}_i - m_j\right\|^p}\right] \left\|\mathbf{x}_i - \mathbf{m}_k\right\|^n \\ &= \left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^{n-p} + \sum_{j \neq k*} \frac{\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^n}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^p} \end{aligned} \tag{5}$$

To create a learning rule and find the new locations of the prototypes that give iteratively better performance, we need to find the partial derivative of the performance function with respect to $\mathbf{m}_{k*}$, which is the closest prototype to $\mathbf{x}_i$, and with respect to $\mathbf{m}_j$ which represent the prototypes that are not the closest to $\mathbf{x}_i$.

$$\begin{aligned} \frac{\partial J_1(\mathbf{x}_i)}{\partial \mathbf{m}_{k*}} &= -(n-p)(\mathbf{x}_i - \mathbf{m}_{k*})\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^{n-p-2} - \\ &\quad n(\mathbf{x}_i - \mathbf{m}_{k*})\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^{n-2} \sum_{j \neq k*} \frac{1}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^p} \\ &= (\mathbf{x}_i - \mathbf{m}_{k*})\, a_{ik*} \end{aligned} \tag{6}$$

$$\frac{\partial J_1(\mathbf{x}_i)}{\partial \mathbf{m}_j} = p(\mathbf{x}_i - \mathbf{m}_j)\frac{\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^n}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^{p+2}} = (\mathbf{x}_i - \mathbf{m}_j) b_{ij}, \quad \forall j \neq k* \tag{7}$$

Solving this over all the dataset results in

$$\mathbf{m}_r(t+1) = \frac{\sum_{i \in V_r} \mathbf{x}_i a_{ir} + \sum_{i \in V_{j, j \neq r}} \mathbf{x}_i b_{ir}}{\sum_{i \in V_r} a_{ir} + \sum_{i \in V_{j, j \neq r}} b_{ir}} \tag{8}$$

where V_r contains the indices of data points that are closest to $\mathbf{m}_r$, V_j contains the indices of all the other points and

$$a_{ir} = -(n-p)\left\|\mathbf{x}_i - \mathbf{m}_r(t)\right\|^{n-p-2} - n\left\|\mathbf{x}_i - \mathbf{m}_r(t)\right\|^{n-2} \sum_{j \neq k*} \frac{1}{\left\|x_i - m_j\right\|^p} \tag{9}$$

$$b_{ir} = p\frac{\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^n}{\left\|\mathbf{x}_i - \mathbf{m}_r(t)\right\|^{p+2}} \tag{10}$$

Using Bregman technique in the iterative learning process, (9) and (10) become:

$$a_{ir} = -(n-p)d_F\left(\mathbf{x}_i, \mathbf{m}_r(t)\right)^{n-p-2} - nd_F\left(\mathbf{x}_i, \mathbf{m}_r(t)\right)^{n-2}\sum_{j\neq k}\frac{1}{d_F\left(\mathbf{x}_i, \mathbf{m}_j\right)^p} \tag{11}$$

$$b_{ir} = p\frac{d_F\left(\mathbf{x}_i, \mathbf{m}_{k*}\right)^n}{d_F\left(\mathbf{x}_i, \mathbf{m}_{r(t)}\right)^{p+2}} \tag{12}$$

where d_F is a Bregman divergence function. For the simulations, we have used $F(x) = 2x \ln x$, which gives $d_F(x, y) = 2(x \ln(x / y) - x + y)$.

In practice, we have found that a viable algorithm may be found by using (12) for all prototypes [and thus never using (11) for the closest prototype]. We will call this the Bregman inverse weighted k-means (BIWK) algorithm.

3.2 Bregman inverse exponential k-means algorithm (BIEK)

In this section, we develop a new algorithm that uses two separate performance functions simultaneuously instead of using one. The first performance function is the minimum function which is used for k-means.

$$J_K = \sum_{i=1}^{N}\min_{j=1}^{K}\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^2 \tag{13}$$

The minimum function is used as in k-means to update the prototypes based on the data samples assigned to the prototype and without considering the whole dataset. The following performance function is designed to help updating the prototypes based on the whole dataset, not only on the members assigned to the prototype.

$$\begin{aligned} J_2 &= \sum_{i=1}^{N}\left[\sum_{j\neq k*}^{K}\frac{1}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|}\right]\left(1 - \exp\left(-\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^3\right)\right) \\ k* &= \arg\min_{k=1}^{K}\left(\left\|\mathbf{x}_i - \mathbf{m}_k\right\|\right) \end{aligned} \tag{14}$$

This performance function deals with the prototypes that are not detected by the minimum function and hence it solves the problem of sensitivity in k-means. Two sets of updates will be derived from these two performance functions, as we will see in the optimisation and implementation section. In the learning process, if the updates prototype is the closest to $\mathbf{x}_i$, we use the first equation that is derived from the minimum performance function in updating, otherwise, we use the second derived one. We accumulate this over the whole dataset. In this way, we make the new algorithm more robust to the initialisation conditions.

3.2.1 Optimisation and implementation

To derive the new clustering algorithm, we need to find the partial derivative of (13) with respect to $\mathbf{m}_k$ and the partial derivative of (14) with respect to $\mathbf{m}_j$ where $\mathbf{m}_k$ represents the closest prototype to $\mathbf{x}_i$, and $\mathbf{m}_j$ represents the other prototypes.

$$\frac{\partial J_K}{\partial \mathbf{m}_k} = \sum_{i \in V_k} -2\left(\mathbf{x}_i - \mathbf{m}_k\right) \tag{15a}$$

where V_k contains the indices of data points that are closest to $\mathbf{m}_k$

$$\frac{\partial J_K}{\partial \mathbf{m}_k} = 0 \tag{15b}$$

$$\sum_{i \in V_k} -2\left(\mathbf{x}_i - \mathbf{m}_k\right) = 0 \tag{15c}$$

$$\mathbf{m}_k = \frac{\sum_{i \in V_k} \mathbf{x}_i}{N_r} \tag{15d}$$

$$\mathbf{m}_k = \frac{\sum_{i \in V_k} \mathbf{x}_i a_{ik}}{\sum_{i \in V_k} a_{ik}} \tag{15e}$$

where N_r is the number of data points that are closest to $\mathbf{m}_k$, $a_{ik} = 1$.

The second performance function provides new calculations for the prototypes that are not closest to data points, and distributes them in a good way to identify the clusters.

$$\frac{\partial J_2}{\partial \mathbf{m}_j} = \sum_{i \in V_d} \frac{\left(\mathbf{x}_i - \mathbf{m}_j\right)\left(1 - \exp\left(-\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^3\right)\right)}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^3} \tag{16a}$$

$$= \sum_{i \in V_d} \frac{\left(\mathbf{x}_i - \mathbf{m}_j\right) c_i}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^3} \tag{16b}$$

where V_d contains the indices of data points that are not closest to $\mathbf{m}_j$

$$c_i = 1 - \exp\left(-\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^3\right) \tag{16c}$$

By assigning the partial derivative to zero and solving for $\mathbf{m}_j$ we have:

$$\frac{\partial J_2}{\partial \mathbf{m}_j} = 0 \tag{16d}$$

$$\Leftrightarrow \sum_{i \in V_d} \frac{\left(\mathbf{x}_i - \mathbf{m}_j\right) c_i}{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^3} = 0 \tag{16e}$$

$$\Leftrightarrow \sum_{i \in V_d} \frac{x_i c_i}{\left\| \mathbf{x}_i - \mathbf{m}_j \right\|^3} = \sum_{i \in V_d} \frac{\mathbf{m}_j c_i}{\left\| \mathbf{x}_i - \mathbf{m}_j \right\|^3} \tag{16f}$$

$$\mathbf{m}_j (t+1) = \frac{\sum_{i \in V_d} \frac{\mathbf{x}_i c_i}{\left\| \mathbf{x}_i - \mathbf{m}_j (t) \right\|^3}}{\sum_{i \in V_d} \frac{c_i}{\left\| \mathbf{x}_i - \mathbf{m}_j (t) \right\|^3}} \tag{16g}$$

$$\text{or } \mathbf{m}_j (t+1) = \frac{\sum_{i \in V_d} \mathbf{x}_i b_{ij}}{\sum_{i \in V_d} b_{ij}} \tag{16h}$$

where

$$b_{ij} = \frac{c_i}{\left\| \mathbf{x}_i - \mathbf{m}_j (t) \right\|^3} \tag{16i}$$

$$= \frac{1 - \exp\left(- \left\| \mathbf{x}_i - \mathbf{m}_{k*} \right\|^3 \right)}{\left\| \mathbf{x}_i - \mathbf{m}_j (t) \right\|^3} \tag{16j}$$

The new locations of all prototypes can be calculated by:

$$\mathbf{m}_r (t+1) = \frac{\sum_{i \in V_r} \mathbf{x}_i a_{ir} + \sum_{i \in V_{j, j \neq r}} \mathbf{x}_i b_{ir}}{\sum_{i \in V_r} a_{ir} + \sum_{i \in V_{j, j \neq r}} b_{ir}} \tag{17}$$

where V_r contains the indices of data points that are closest to $\mathbf{m}_r$, V_j contains the indices of all the other points and

$$a_{ir} = 1 \tag{18}$$

$$b_{ir} = \frac{1 - \exp\left(- \left\| \mathbf{x}_i - \mathbf{m}_{k*} \right\|^3 \right.}{\left\| x_i - m_r (t) \right\|^3} \tag{19}$$

By applying a Bregman divergence to (18) and (19) we get:

$$a_{ir} = 1 \tag{20}$$

$$b_{ir} = \frac{1 - \exp\left(- d_F \left(\mathbf{x}_i , \mathbf{m}_{k*} \right)^3 \right)}{d_F \left(\mathbf{x}_i , \mathbf{m}_r (t) \right)^3} \tag{21}$$

In execution, we have found that the main computational cost comes from (21). Assume we have K prototypes and N data points, and then as every data point is closest to one and only one prototype, for one iteration we have N * K loops, and in each loop either (20) or (21) will be executed.

(20) will be executed N times while (21) will be executed N(K – 1) times.

It is possible to find a viable algorithm by using (21) for all prototypes [and thus never using (20) for the closest prototype].

3.3 The BIWC algorithm

Consider the following performance function:

$$J_3 = \sum_{i=1}^{N}\sum_{k=1}^{K}\frac{1}{\left\|\mathbf{x}_i - \mathbf{m}_k\right\|^P} \tag{22}$$

$$\frac{\partial J_3}{\partial \mathbf{m}_k} = \sum_{i=1}^{N} P\left(\mathbf{x}_i - \mathbf{m}_k\right)\frac{1}{\left\|\mathbf{x}_i - \mathbf{m}_k\right\|^{P+2}} \tag{23}$$

$$\frac{\partial J_3}{\partial \mathbf{m}_k} = 0 \Rightarrow \mathbf{m}_k = \frac{\sum_{i=1}^{N} b_{ik}\mathbf{x}_i}{\sum_{i=1}^{N} b_{ik}} \text{ where } b_{ik} = \frac{1}{\left\|\mathbf{x}_i - \mathbf{m}_k\right\|^{P+2}} \tag{24}$$

The partial derivative of J_3 with respect to $\mathbf{m}_k$ can be used to maximise the performance function J_3. So the implementation of (24) will always move $\mathbf{m}_k$ to the closest data point to maximise J_3 to ∞.

However, the implementation of (24) will not identify any clusters as the prototypes always move to the closest data point. But the advantage of this performance function is that it does not leave any prototype far from data: all the prototypes converge toward the data. This performance function moves each prototype toward one of the closest data points to maximise J_3 to ∞. In this way, we will not have any dead prototypes even if the prototypes are initialised far from the data points.

We can enhance this algorithm's ability to identify the clusters without losing its property of pushing the prototypes inside data clusters by changing b_{ik} in (24) to the following:

$$b_{ik} = \frac{\left\|\mathbf{x}_i - \mathbf{m}_{k*}\right\|^{P+2}}{\left\|\mathbf{x}_i - \mathbf{m}_k\right\|^{P+2}} \tag{25}$$

where $\mathbf{m}_{k*}$ is the closest prototype to $\mathbf{x}_i$.

By applying Bregman divergence, we get:

$$b_{ik} = \frac{d_F\left(\mathbf{x}_i, \mathbf{m}_{k*}\right)^{P+2}}{d_F\left(\mathbf{x}_i, \mathbf{m}_k\right)^{P+2}} \tag{26}$$

where $\mathbf{m}_{k*}$ is the closest prototype to $\mathbf{x}_i$.

With this change, we have an interesting behaviour: (26) works to maximise J_3 by moving the prototypes to the free data points (or clusters) instead of the closest data point (or local cluster).

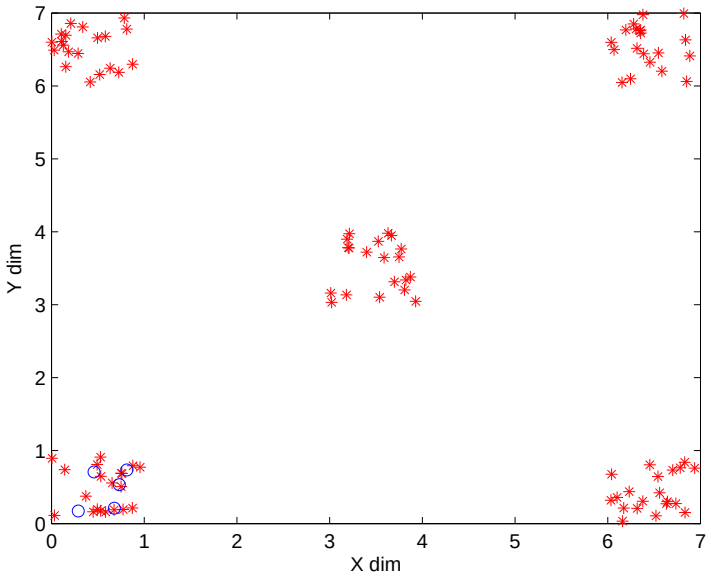
We will call this the Bregman inverse weighted clustering (BIWC) algorithm, which is a special case of the BIWK algorithm.

3.4 Simulations

3.4.1 Artificial dataset

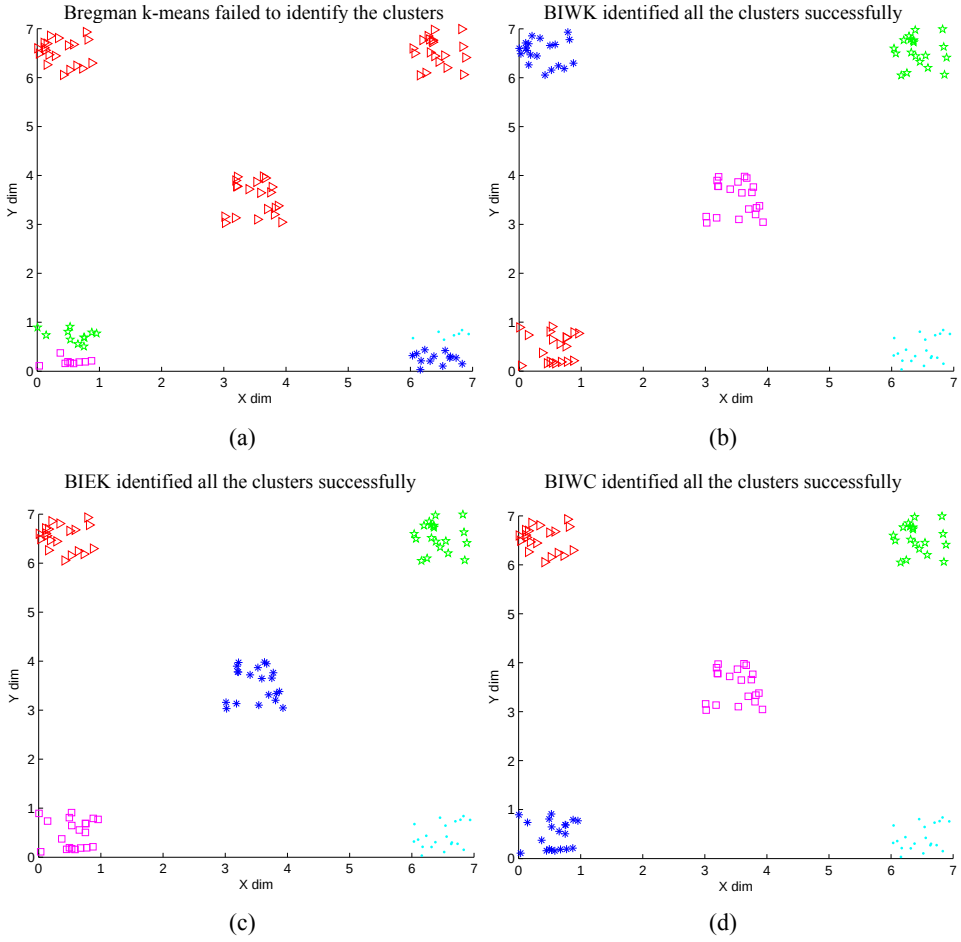
We illustrate these new algorithms with a few simulations on artificial two dimensional datasets, since the results are easily verified visually. In Figure 1, we have an artificial dataset consisting of 100 data points in five clusters. We have generated this dataset using Matlab with five clusters: each sample is drawn independently from a uniform distribution within the square shown. Four clusters are located at corners of a large square while the fifth is in the centre of the data. Figure 2 shows the results after applying Bregman k-means, BIWK, BIEK and BIWC respectively to the artificial dataset. In Figure 1, the prototypes have all been initialised within a single cluster. As shown in Figure 2, while the Bregman k-means algorithm failed to identify the clusters, the new Bregman algorithms BIWK, BIEK and BIWC identified them successfully.

Figure 1 Artificial dataset – 100 data points in five clusters (see online version for colours)



Note: Dataset is shown as five clusters of red '*'s, prototypes are initialised to lie within one cluster and shown as blue 'o's.

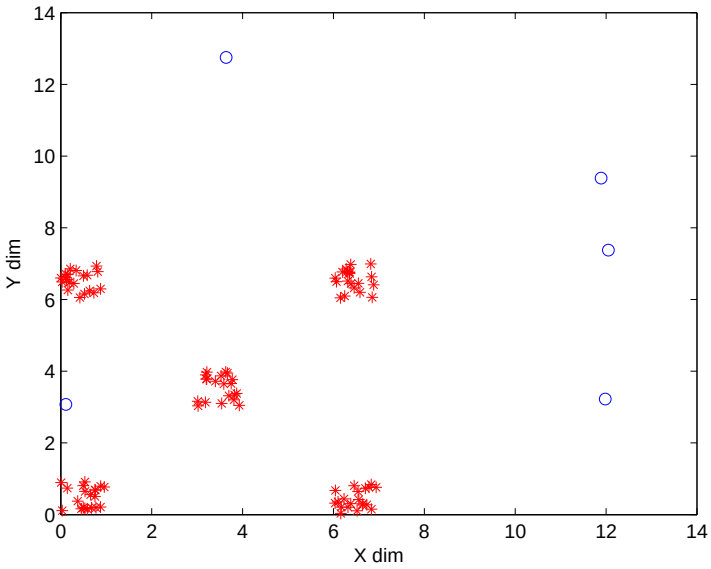
Figure 2 (a) Bregman k-means result (b) BIWK result (c) BIEK result (d) BIWC result (see online version for colours)



In Figure 3, we have used the same artificial dataset as in Figure 1, but with different prototypes initialisation. We have initialised the prototypes randomly around the data samples. We have applied the Bregman k-means and the new Bregman clustering algorithms to the data in Figure 3. The results are shown in Figure 4. As shown in Figure 4(a), the Bregman k-means failed to identify all the clusters successfully. Instead, it has merged some clusters together. Also, Bregman k-means has failed to activate the dead prototypes. On the other hand, the new Bregman clustering algorithms BIWK, BIEK and BIWC have identified all the clusters successfully.

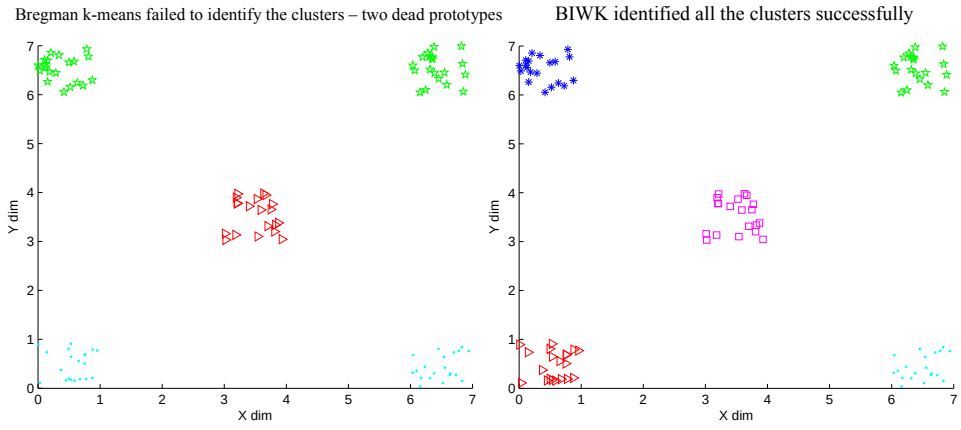
To measure the strength of the new algorithms in finding clusters, we have applied the Bregman k-means, BIWK, BIEK and BIWC to the dataset shown in Figure 5. In this figure, we have initialised 100 data points randomly. Each data point represents a cluster. We have initialised 100 prototypes randomly inside the data. As shown in Figure 6, Bregman k-means failed to identify all the clusters. Also, there are 87 dead prototypes and Bregman k-means failed to activate them to be able to find the missing clusters. While the new Bregman algorithms identified all the clusters successfully.

Figure 3 Artificial dataset – 100 data points in five clusters – bad initialisation (see online version for colours)



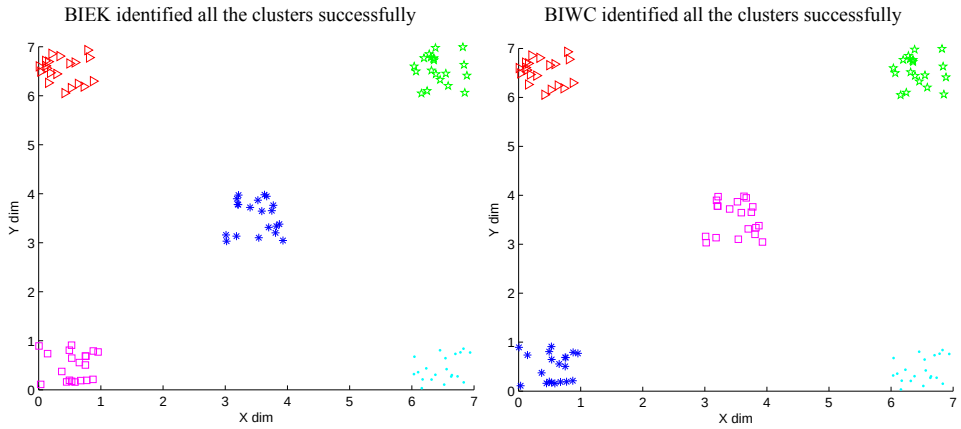
Note: Dataset is shown as five clusters of red '*'s, prototypes are initialised randomly around the data and shown as blue 'o's.

Figure 4 (a) Bregman k-means result (b) BIWK result (c) BIEK result (d) BIWC result (see online version for colours)



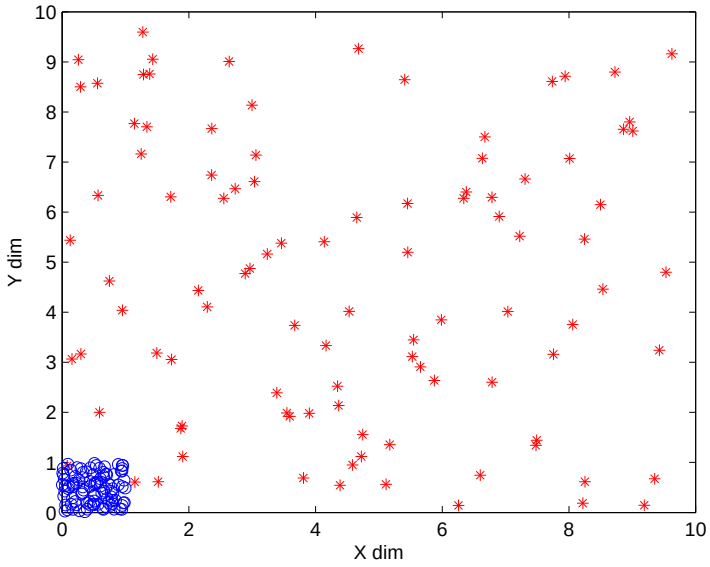
(a) (b)

Figure 4 (a) Bregman k-means result (b) BIWK result (c) BIEK result (d) BIWC result (continued) (see online version for colours)



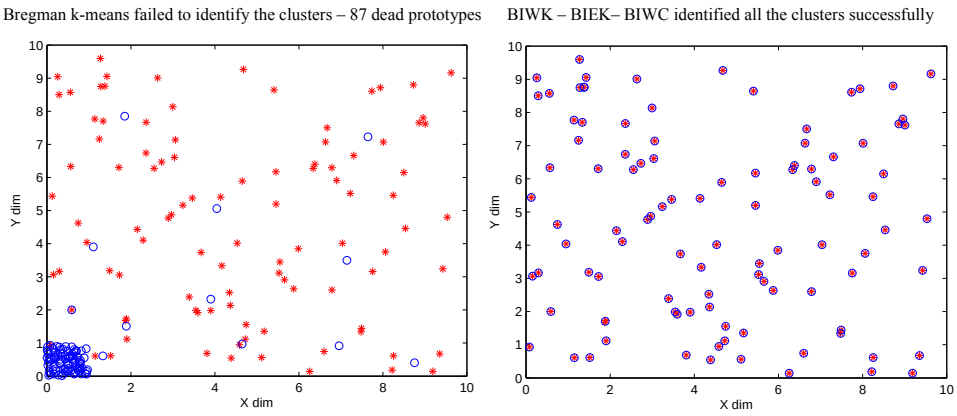
(c) (d)

Figure 5 Artificial dataset – 100 points in 100 clusters – all prototypes are initialised inside data (see online version for colours)



Note: 100 data points (clusters) are shown as red '*'s and 100 initialised prototypes are shown as blue 'o's.

Figure 6 (a) Bregman k-means result (b) BIWK, BIEK and BIWC result (see online version for colours)



(a) (b)

3.4.2 Real datasets

The iris[1] dataset is a real dataset with 150 random samples of flowers from the iris species setosa, versicolor, and virginica collected by Anderson in 1935 (Anderson, 1935). There are 50 observations from each species for sepal length, sepal width, petal length and petal width in cm. This dataset was used by Fisher (1936) in his initiation of the linear discriminant-function technique.

The glass[2] dataset is another real dataset which has 214 samples with ten dimensions and six types.

In the algae[3] dataset, we have 72 samples classified into nine types. Each sample is recorded as an 18 dimensional vector representing the magnitudes of various pigments.

The genes[4] dataset has 40 samples with 3,036 dimensions and three types of bladder cancer. This dataset is therefore extremely high dimensional but with rather few samples.

We now apply Bregman k-means, BIWK, BIEK, and BIWC algorithms to the iris, glass, genes and algae real datasets. We run each algorithm with ten different prototypes' initialisations. Each time we calculate the quantisation error (Qe). The number of initialised prototypes equals the number of clusters in the real dataset. Table 1 shows the results on iris dataset, Table 2 shows the results on glass dataset, Table 3 shows the results on genes dataset and Table 4 shows the results on algae dataset.

In Table 1, we can see that all the new algorithms give the same value. The reason for that is that the iris dataset consists of three classes, one of which is far away and easy to identify with the new algorithms. There is a small amount of interference between the other two classes. Our algorithms are robust to the initialisation of the prototypes and can find the optimal solution easily for the iris dataset. Thus, we can see the new algorithms give the same value for this dataset. In addition, k-means sometimes finds the optimal solution and sometimes fails, due to its sensitivity to the prototypes' initialisations.

From Tables 1, 2, 3 and 4, we see that the new algorithms work better and more robustly in general than Bregman k-means and give smaller quantisation error. BIWC appears to be the most robust of the algorithms while in the glass dataset in particular, BIWK is almost as sensitive as standard Bregman k-means. Also, we see that all the new

algorithms are comparable to each other. Thus, we cannot say this algorithm is better than that; any one of them can be the best, depending on the dataset we use. However, we can see that the BIWC is more robust with respect to initialisation than the other algorithms.

In the genes dataset, we have 3,036 dimensions, thus initial locations of the prototypes are normally very far from clusters and this may affect badly the clustering algorithms. However, as shown in Table 3 the new Bregman algorithms are still robust to the initial conditions and give good results, while Bregman k-means got the worst results compared to the other datasets.

Table 1 Results of applying Bregman k-means, BIWK, BIEK and BIWC to the iris dataset

	Bregman k-means *Qe*	*BIWK* *Qe*	*BIEK* *Qe*	*BIWC* *Qe*
1	39	22	22	22
2	22	22	22	22
3	39	22	22	22
4	22	22	22	22
5	22	22	22	22
6	39	22	22	22
7	39	22	22	22
8	22	22	22	22
9	22	22	22	22
10	39	22	22	22
Avg.	30.5	22	22	22

Notes: The rows show the results of applying the algorithms with different prototypes' initialisation. Last row shows the average of quantisation error and number of errors for each algorithm. For each algorithm, we have one column shows the quantisation error.

Table 2 Results of applying Bregman k-means, BIWK, BIEK and BIWC to the glass dataset

	Bregman k-means *Qe*	*BIWK* *Qe*	*BIEK* *Qe*	*BIWC* *Qe*
1	875	836	817	812
2	990	824	817	812
3	929	825	817	812
4	842	938	818	812
5	805	940	823	812
6	804	803	817	812
7	847	891	811	812
8	871	819	817	812
9	837	818	825	812
10	825	836	817	812
Avg.	862.5	853	817.9	812

Notes: The rows show the results of applying the algorithms with different prototypes' initialisation. Last row shows the average of quantisation error and number of errors for each algorithm. For each algorithm, we have one column showing the quantisation error.

Table 3 Results of applying Bregman k-means, BIWK, BIEK and BIWC to the genes dataset

	Bregman k-means *Qe*	*BIWK* *Qe*	*BIEK* *Qe*	*BIWC* *Qe*
1	242,400	217,640	244,490	231,030
2	396,410	233,700	226,770	231,030
3	396,410	232,310	244,490	231,030
4	217,490	224,470	227,210	244,840
5	216,180	227,910	226,420	244,840
6	396410	226,440	244,490	231,030
7	398,280	213,910	239,500	244,840
8	244,490	222,490	244,490	231,030
9	282,200	213,910	224,370	231,030
10	396,410	217,880	244,490	244,840
Avg.	318,668	223,066	236,672	236,552

Notes: The rows show the results of applying the algorithms with different prototypes' initialisation. Last row shows the average of quantisation error and number of errors for each algorithm. For each algorithm, we have one column showing the quantisation error.

Table 4 Results of applying Bregman k-means, BIWK, BIEK and BIWC to the algae dataset

	Bregman k-means *Qe*	*BIWK* *Qe*	*BIEK* *Qe*	*BIWC* *Qe*
1	38	22	20	15
2	34	24	19	15
3	19	15	23	15
4	27	20	28	20
5	30	24	19	16
6	19	17	19	16
7	46	21	18	16
8	26	17	19	20
9	18	18	19	16
10	28	26	20	15
Avg.	28.5	20.4	20.4	16.4

Notes: The rows show the results of applying the algorithms with different prototypes' initialisation. Last row shows the average of quantisation error and number of errors for each algorithm. For each algorithm, we have one column showing the quantisation error.

4 Conclusions

We have introduced three new algorithms as a family of Bregman clustering algorithms. They were all developed with the same underlying rationale: they are designed to improve upon Bregman k-means' convergence to local optima depending on the initial

conditions with which we start the algorithm. Also each of the algorithms incorporates both global and local knowledge so that the means when being re-positioned take account of the positions of all the other means as well as the data samples.

The Bregman inverse exponential weighted k-means exhibits very similar behaviour to the BIWK in that all prototypes are sensibly located. This algorithm may be thought to use a similar technique to the standard soft k-means [MacKay, (2003), p.289] though the Bregman inverse exponential weighted k-means is much better at solving the problem of sensitivity to initial conditions.

The BIWC algorithm is a special case of the BIWK, in which we use only one set of updates (21). All the algorithms, except the BIWC algorithm which is special case of BIWK, have two sets of updates rather than a single update for all prototypes. This is a symmetry-breaking factor which enables local minima to be avoided, even if all prototypes are initialised in the same location as one point.

In this paper, we have shown the results of right Bregman divergences, i.e., the centroids are in the right position of the divergence. We have also results using the left divergence and results on standard datasets used for clustering comparisons.

References

Anderson, E. (1935) 'The irises of the Gaspe peninsula', *Bulletin of the American Iris Society*, Vol. 59, pp.2–5.

Arai, K. and Barakbah, A.R. (2007) 'Hierarchical k-means: an algorithm for centroids initialization for k-means', Reports of the Faculty of Science and Engineering Saga University, Vol. 36, No. 1, pp.25–31.

Arthur, D. and Vassilvitskii, S. (2007) 'k-means++: the advantages of careful seeding', *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.1027–1035.

Banerjee, A., Meruga, S., Dhillon, I. and Ghosh, J. (2005) 'Clustering with Bregman divergences', *Journal of Machine Learning Research*, Vol. 6, pp.1705–1749.

Barbakh, W. and Fyfe, C. (2008a) 'Local vs global interactions in clustering algorithms', *International Journal of Knowledge Based Intelligent Engineering Systems*, Vol. 10, No. 2, pp.83–99.

Barbakh, W. and Fyfe, C. (2008b) 'Online clustering algorithms', *International Journal of Neural Systems*, Vol. 18, No. 3, pp.1–10.

Cao, F., Liang, J. and Jiang, G. (2009) 'An initialization method for the k-means algorithm using neighborhood model', *Computers and Mathematics with Applications*, August, Vol. 58, pp.474–483.

Collins, M., Dasgupta, S. and Shapire, R.E. (2001) 'A generalization of principal component analysis to the exponential family', *Nips*, Vol. 14, pp.617–624.

Dhillon, I.S., Mallela, S. and Kumar, R. (2003) 'A divisive information theoretic feature clustering algorithm for text classificationl', *Journal of Machine Learning Research*, Vol. 3, pp.1265–1287.

Fisher, R.A. (1936) 'The use of multiple measurements in taxonomic problems', *Annals of Eugenics*, Vol. 7, pp.179–188.

Frigyik, B.A., Srivastava, S. and Gupta, M.R. (2008) 'Functional Bregman divergence and Bayesian estimation of distributions', *IEEE Transactions on Information Theory*, Vol. 54, No. 11, pp.5130–5139.

Hartigan, J. and Wang, M. (1979) 'A k-means clustering algorithm', *Applied Statistics*, Vol. 28, pp.100–108.

Khan, S. and Ahmad, A. (2004) 'Cluster center initialization algorithm for k-means clustering', *Pattern Recognition Letters*, August, Vol. 25, pp.1293–1302.

Lloyd, S.P. (1982) 'Least squares quantization in pcm', *IEEE Transactions on Information Theory*, Vol. 28, No. 2, pp.128–137.

Lu, J.F., Tang, J.B., Tang, Z.M. and Yang, J.Y. (2008) 'Hierarchical initialization approach for k-means clustering', *Pattern Recognition Letters*, April, Vol. 29, pp.787–795.

MacKay, D.J. (2003) *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, UK.

MacQueen, J. (1967) 'Some methods for classification and analysis of multivariate observations', *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp.281–297.

Neilsen, F. and Nock, R. (2007) 'On the centroids of symmetrized Bregman divergences', *CoRR*, abs/0711.3242, available at http://arxiv.org/abs/0711.3242 (accessed on 14/11/2011).

Neilsen, F., Boissonnat, J-D. and Nock, R. (2007a) 'Bregman voronoi diagrams: properties, algorithms and applications', available at http://arxiv.org/abs/0709.2196 (accessed on 14/11/2011).

Neilsen, F., Boissonnat, J-D. and Nock, R. (2007b) 'On Bregman voronoi diagrams', *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp.746–755.

Wang, X. and Fyfe, C. (2010) 'Independent component analysis using Bregman divergences', *The Twenty Third International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, IEA-AIE 2010*, June, Cordoba, Spain, pp.627–636.

Wang, X., Crowe, M. and Fyfe, C. (2011) 'Dual stream data exploration', *International Journal of Data Mining, Modelling and Management*, Vol. 44, No. 5, pp.1137–1154.

Notes

1 The iris dataset is available at http://mlearn.ics.uci.edu/databases/.

2 The glass dataset is available at http://mlearn.ics.uci.edu/databases/.

3 The algae dataset is available at http://mlearn.ics.uci.edu/databases/.

4 The genes dataset is available at http://www.ihes.fr/~zinovyev/princmanif2006/.