10.1590/s1982-21702018000400027

ARTICLE

EVALUATION OF THE CLUSTERING PERFORMANCE OF AFFINITY PROPAGATION ALGORITHM CONSIDERING THE INFLUENCE OF PREFERENCE PARAMETER AND DAMPING FACTOR

Avaliação do Desempenho da Clusterização do Algoritmo Propagação de Afinidade Considerando a Influência do Parâmetro de Preferência e do Fator de Amortecimento

André Fenias Moiane[1] - ORCID: 0000-0001-8242-6264

Álvaro Muriel Lima Machado[1] - ORCID: 0000-0002-7371-3461

[1] Universidade Federal do Paraná, Departamento de Geomática, Curitiba, Paraná, Brasil.
E-mail: andremoiane16@gmail.com; alvaromlmachado@gmail.com

Received in July 24th 2016

Accepted in May 04th 2018

Abstract:

The identification of significant underlying data patterns such as image composition and spatial arrangements is fundamental in remote sensing tasks. Therefore, the development of an effective approach for information extraction is crucial to achieve this goal. Affinity propagation (AP) algorithm is a novel powerful technique with the ability of handling with unusual data, containing both categorical and numerical attributes. However, AP has some limitations related to the choice of initial preference parameter, occurrence of oscillations and processing of large data sets. This paper evaluates the clustering performance of AP algorithm taking into account the influence of preference parameter and damping factor. The study was conducted considering the AP algorithm, the adaptive AP and partition AP. According to the experiments, the choice of preference and damping greatly influences on the quality and the final number of clusters.

Keywords: Affinity Propagation; Clustering; Accuracy; Preference; Damping Factor.

How to cite this article: MOIANE, A. F. and MACHADO, A.M.L. Evaluation of the Clustering Performance of Affinity Propagation Algorithm Considering the Influence of Preference Parameter and Damping Factor. *Bulletin of Geodetic Sciences*, Vol. 24 (4): 426-441, Oct-Dec, 2018.

Resumo:

A identificação de padrões de dados subjacentes significativos tais como composição de imagem e arranjos espaciais é crucial em tarefas de sensoriamento remoto. Porém, o desenvolvimento de uma abordagem eficaz para extração de informações é crucial para alcançar esse objetivo. O algoritmo propagação de afinidade (PA) é uma técnica nova e poderosa capaz de lidar com dados incomuns, contendo atributos numéricos e/ou categóricos. No entanto, PA tem algumas limitações relacionadas com a escolha do parâmetro preferência inicial, ocorrência de oscilações e processamento de grandes conjuntos de dados. Este artigo avalia o desempenho da clusterização do algoritmo PA levando em consideração a influência do parâmetro preferência e o fator de amortecimento. O estudo foi realizado considerando o algoritmo PA, PA adaptativa e PA por partição. De acordo com os experimentos, a escolha da preferência e do fator de amortecimento influenciam grandemente no número de clusters final e na qualidade da clusterização.

Palavras-chave: Propagação de Afinidade; Clusterização; Acurácia; Preferência; Fator de Amortecimento.

1. Introduction

Data clustering is one of the fundamental tasks in remote sensing, used for information extraction and classification purposes (Dermoudy *et al*. 2009). It aims at partitioning datasets into several clusters, and such that similar data points remain in the same group and dissimilar to different ones (Driver and Kroeber 1932; Tryon and Robert 1939; Cattell 1943; Soman and Ali 2015). In most simple terms, data clustering can be considered as a method to identify significant underlying patterns in data, referring to statistical distributions followed by distinct classes to which the data can be categorized (Duda and Hart 1973; Jain and Dubes 1988; Galdi *et al*. 2014).
In remote sensing, clustering algorithms can be applied in unsupervised classification to divide multispectral and hyperspectral spaces for extraction of patterns associated with land-cover classes (Dey *et al.* 2010; Chehdi *et al.* 2014). These algorithms can also be used as a pre-processing step before performing any classification task (Dermoudy *et al*. 2009).

Researchers have made various attempts to develop an effective tool for information extraction and image classification. Due to the simplicity of implementation in a variety of scenarios, clustering algorithms based on distance are widely used and researched (Batra 2011). The main limitation of these methodologies is the selection of a proper similarity metric that can be able to distinguish similar from dissimilar data points without supervision. Thus, the clustering problem can be simplified in finding a distance metric for the used data type (Aggarwal 2003; Aggarwal and Reddy 2013). Traditional clustering techniques compute similarities between data points using Euclidean distance, which is suitable for purely numeric data sets (Duda and Hart 1973). Actually, real world applications are complex; most of datasets are mixed containing both numeric and categorical attributes, what makes the Euclidean distance function to fail in judging the similarity between two data points (Zhang and Gu 2014).

Affinity Propagation (AP) was studied by Frey and Dueck, and described as a powerful clustering methodology which propagates messages of affinities between pairwise points in a factor graph (Frey and Dueck 2007). Compared to the traditional approaches, AP technique can also use

nonmetric similarities as input data, making the data analysis exploration suitable for unusual metrics of similarity (Guan *et al.* 2011).

Several studies related to AP have been performed. Adaptive AP was conducted by Wang *et al*. (2008), fast sparse AP (Jia *et al*. 2008), binary variable model AP (Givoni and Frey 2009), relational AP (Plangprasopchok *et al*. 2010), fuzzy statistic AP (Yang *et al*. 2010), fast AP (Fanhua *et al*. 2012), landmark AP (Thavikulwat 2014), improved adaptive AP (Zhou *et al*. 2015), modified AP (Serdah and Ashour 2016) and many more emerged since AP was introduced. Although AP provides satisfactory results, it still presents limitations related to its performance, such as the choice of initial preference parameter, occurrence of oscillations and processing of large data sets.

To address the above mentioned issues, more investigation on AP variants needs to be undertaken. However, based on the literature reviewed, no related study was performed to evaluate how the preference parameter and damping factor influence in the AP clustering performance. Studies conducted by Bombatkar and Parvat (2015); Refianti and Mutiara (2016) and Galdi *et al.* (2014) presented clustering techniques, but, none of them performed as this study suggests. Thus, by knowing to what extent these parameters influence in the clustering performance of AP algorithm, the proposed study would contribute for improvement of AP and possibly for development of new AP variants.

2. Affinity Propagation Algorithms and Variants

2.1 Affinity Propagation

Affinity propagation (AP) is an algorithm that identifies centres of clusters, also called *exemplars* to form its clusters around them. This algorithm simultaneously considers all the points in the set as probable candidates to become centres of the clusters and propagate exchanges of messages between the points until the emergence of good exemplars and clusters (Frey and Dueck 2007).

AP uses as input real-valued similarities $S(i,j)$, describing how well the j-th point is appropriated to become an exemplar for the i-th point. When the points lay along the matrix diagonal, i.e., $i = j$, the similarity matrix $S(i,j)$ is called *preference*, and indicates how probable the i-th point is to be selected as an exemplar. Preferences can be set to a global value, or for particular data points.

High preference values will cause AP to find many clusters, while low values will lead to a small number of clusters. A good initial choice to determine the preference is to take the minimum or the median similarities. The similarity is commonly expressed as a negative squared Euclidean distance according to equation (1), in which the parameters x_i and x_j are the positions of data points i and j in 2D space (Dueck 2009).

$$S(i, j) = -\left\| x_i - x_j \right\|^2 \tag{1}$$

The number of defined centres of clusters is mainly influenced by the values of preference, but it either emerges from the message exchanging process shown in the factor graph of Figure 1. A factor graph is defined as a bi-partite graph consisting of a set of nodes representing random variables and a set of functions. This graphical model represents global functions or probability

distributions that can be factored into simpler local functions. According to Figure 1, each component of *F(c;s)* is represented by a function node and each c_i is represented by a variable node. Each $f_k(c)$ term appearing in the graph has a corresponding function node that is connected to all variables $c_1, c_2,\ldots,c_N$. In addition, each $s(i,c_i)$ term has a corresponding function node that is connected to a single variable c_i. The log of the global function *F(c;s)*, in this case *S(c)* is referred to as net similarity *S(i,j)*, and is obtained by summing together all the log-functions on the nodes (Dueck 2009).

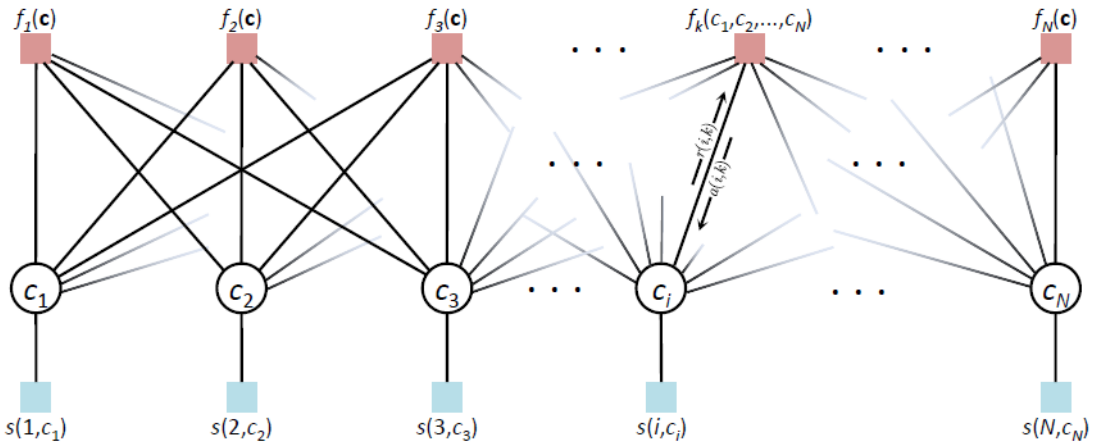

Figure 1: Affinity Propagation Factor graph (Frey and Dueck 2007).

The process of sending messages is presented in Figure 2. In the figure, availability and responsibility messages are exchanged. *Responsibilities are* sent from data point *i* to candidate exemplar point *k*, and show how evident point *k* is to be an exemplar for point *i*, counting with other potential exemplars for point *i*. A*vailabilities*, are sent from candidate exemplar point *k* to point *i*, and show the chance the point *k* as to be selected as its exemplar, considering the support the other points give (Dueck 2009).

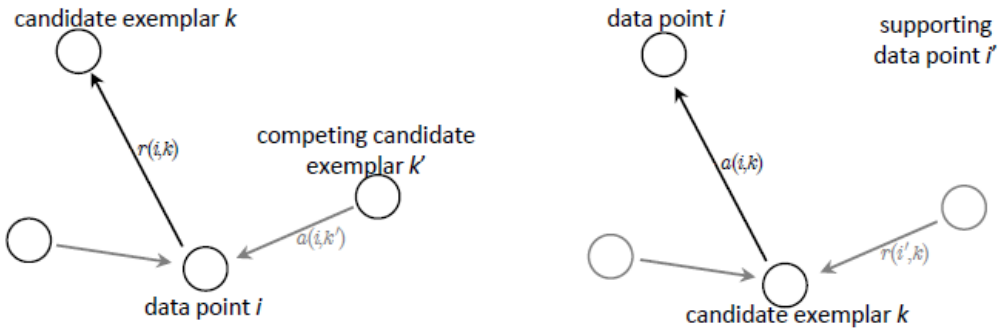

Figure 2: Propagation of two messages between data points: (left) "*responsibilities*" *r(i,k)* are sent for data point *i* to candidate exemplar *k*, and (right) "*availabilities*" *a(i,k)* are sent from candidate exemplar *k* to data point *i* (Dueck 2009).

The responsibility and availability values are adjusted as follows:

$$\forall_{i,k} : r(i,k) = S(i,k) - \max_{k^{'}:k^{'}\neq k}\left[S(i,k^{'}) + a(i,k^{'})\right] \tag{2}$$

In the equation (2), the letter *i* represents a data point and *k'* stands for a competing candidate exemplar. In the first iteration, because the availabilities are initialized to zero, *r(i,k)* is set to the input similarity between point *i* and point *k* as its exemplar, minus the maximum of the similarities between point *i* and other candidate exemplars *k'*.

$$\begin{aligned} &\forall_{i,k} : a(i,k) = \sum \max\left[0, r(i^{'},k)\right], \quad for\ k = i, i^{'} : i^{'} \neq i \\ &\forall_{i,k} : a(i,k) = \min\left[0, r(k,k) + \sum_{i^{'}:i^{'}\in\{i,k\}} \max\left[0, r(i^{'},k)\right]\right] \quad for\ k \neq i \end{aligned} \tag{3}$$

In the above equations, the availability *a(i,k)* is set to the self-responsibility *r(k,k)* plus the sum of the positive responsibilities candidate exemplar *k* receives from other supporting points *i'*. Only the positive portions of incoming responsibilities are added, because it is only necessary for a good exemplar to explain some data points well (positive responsibilities), regardless of how poorly it explains other data points (negative responsibilities).

At each iteration, the assignment of items to exemplars is defined as:

$$\phi(x_i) = \arg\max\ k\{r(i,k) + a(i,k)\} \tag{4}$$

In the equation above, $\phi(x_i)$ is the exemplar for data point x_i. The message propagation process stops as soon as it reaches a specified number of iterations or when the cluster structure stabilises with a given number of iterations (Dueck 2009).

2.2 Adaptive Affinity Propagation

Adaptive Affinity Propagation (AAP) algorithm is an AP variant developed to deal with issues related with setting the initial *preference* value *p* which is determined by taking the minimum or the median of the similarity *S(i,j)*, and occurrence of oscillations in the original AP algorithm. The choice of *preference* influences on the final resulting number of clusters, and the oscillation may often cause the process to fail to reach the convergence. AAP includes two parts, adaptive damping and adaptive scape. Adaptive damping aims to adaptively remove numerical fluctuations whereas adaptive scaping is used to adaptively modify *preferences* to gain the optimal clustering. The values of the adaptive factors p_s and λ_s can be determined empirically, being λ_s positive while p_s negative (Sun *et al*. 2009). The initial value used for λ ii this article was 0.5.

According to experiments, and considering that big number of clusters is more sensitive to *ps* than that of small number of clusters, the adaptive decreasing step *ps*=0.01*p*/*q* was designed, where $q = 0.1\sqrt{K+50}$ is the decreasing parameter. Thus, *q* is adjusted dynamically with the number of exemplars *K*, and the *ps* is smaller when *K* is bigger, while the *ps* is larger when *K* is smaller. Although the damping factor λ close to 1 has more probability to eliminate oscillation, the responsibilities and availabilities are updated slowly and much more time is needed to run AP. The better choice is to check the effect of oscillation elimination while increasing λ gradually. Following this idea, the adaptive damping method was designed, first to detect whether oscillations occur in an iteration of the AP, and increase λ once by a step λ_s such as 0.05 if oscillations were detected, otherwise return to the first step. If it fails to remove oscillations by increasing λ to 0.85 or higher, an adaptive escape is used to avoid oscillations, and *p* is gradually decreased by *ps* and λ_s falls to 0.025 until oscillations disappear. If oscillations occur, increasing λ by a step such as 0.05; if λ≥0.85, decreasing *p* by step *ps*, otherwise return to first step of the adaptive damping method.

The pseudo code of the AAP variant is as follows:

Input : Number of Data Points *N*, Similarity Matrix *S(i,j)*
Output : Clusters of each Data Points
1) Execute AP procedure, get the number of clusters *K(i).*
2) If *K(i)* ≤ *K(i + 1)*, then go to step 4. Else, *count* = 0, then go to step 3.
3) $\lambda = \lambda + \lambda_s$, then go to step 1. If $\lambda > 0.85$, then $p = p + p_s$, *S(i,i)* = *p.* Else go to step 1.
When the value of λ is larger than 0.85, $\lambda_s = 0.025$, otherwise $\lambda_s = 0.05$
4) If | *Cmax* - *K(i)* | > *CK,* then A_s = -20* | *K(i)* - *Cmin* |.
Go to step 6. Else, delay 10 iterations and then go to step 5.
5) If *K(i)* ≤ *K(i + 1)*, *count* = *cout* + 1, A_s = *count* *p_s. Go to step 6. Else, go to step 1.
6) $p = p + A_s$, then *S(i,i)* = *p.*
7) If *i* = *maxits* or *K(i)* ≤ *Cmin,* the algorithm terminates. Else, go to step 1.

In the code, *Cmax* and *Cmin* are respectively the expected maximal and minimal number of clusters. *CK* should be a positive integer and can be determined empirically.

2.3 Partition Affinity Propagation

Partition Affinity Propagation (PAP) is an extension of AP that consists in partitioning the similarity matrix into sub-block matrices. This procedure can speed up the AP execution, reduce the number of iterations and maintain the same accuracy as the original AP (Xia *et al*. 2008). If the similarity between data points is finite, what means that this data set is of dense relationship such as most of image data. For this kind of data set the AP message-passing procedure is performed in a dense matrix. The computation overload is directly related with the number of iterations. Thus, for an AP iteration, each element in the matrix of responsibility *r* (*i,k*), must be computed once, and each computation must be applied on 1−*N* elements, being *N* the size of similarity matrix (Zhang *et al*. 2008). The PAP algorithm can be summarized by the following pseudo code:

Input : Number of Data Point *N*, Similarity Matrix *S(i,j)*
Output : Clusters of each Data Points
1) Calculate the Number of Partitions: *k = sqrt(N)/2*
2) Calculate the Matrix Partition Size: *Ps = N/k*
3) Set Partition End point to zero: *Ep = 0*
4) For all the partitions (i = 1 to *k*) calculate:
   Partition Start Point: *Sp = 1+Ps*(i - 1)*
   Matrix Size: *Ps = N – Ps*(k - 1)*
   Partition End Point: *Ep = Ep + Ps*
   Construct Sub Matrix: *SM(i, j) = S(Sp : Ep, Sp : Ep)*
   Run Responsibility Update Once with matrix *SM(i, j)*
   Run Availability Update Once and save in *SubA(i)*
5) Combine All obtained *A(i, j) = SubA(*)*
6) Run Affinity Propagation with New *A(i, j)*.

2.4 Partition Adaptive Affinity Propagation

Partition Adaptive Affinity Propagation (PAAP) is an approach of adaptive affinity propagation that can combine the advantages of PAP and AAP (Sun *et al*. 2009). This algorithm can eliminate message oscillations by using the AAP method. The method consists of two parts, fine adjustment and coarse adjustment. Fine adjustment is used to decrease the values of parameter *preference* at a low rate of speed, and coarse adjustment is employed to quickly decrease the values of *preference* correspondingly. The original similarity matrix is decomposed into sub-matrices to gain higher execution speed, when executing PAAP and can yield optimal clustering solutions on both dense and sparse datasets (Xia *et al*. 2008). The advantages of PAAP are that it cannot only improve the AP algorithm but also automatically adjusts the parameters *preference* and *dumping factor.* The PAAP proceeds the same way as AAP but considers the decomposition of similarity matrix *S (i,j)* into averagely square sub-matrices, and runs AP on them.

3. Methodology

3.1 Data sets and Software

This research aims to evaluate the clustering performance on AAP, PAP in relation to the original AP. The performance evaluation of the chosen AP variants was based on two aspects, temporal and grouping efficiency. The temporal efficiency was measured by the number of iterations and the time the algorithm takes to compute and finish the clustering process, while the grouping efficiency defines the number of clusters, and was measured by Silhouette index according to equations (5), (6) and (7). The experiments were carried out in MATLAB R2013a on a PC computer with following characteristics: Model HP Pavilion dv6 Notebook, Processor on 8GB RAM Intel(R) Core (TM) i7-2630QM CPU@2.00 GHz.

To perform the experiments two types of data sets were used, simulated and real data sets. A two-dimensional simulated data set here referred to Simdata was generated using the random function in MATLAB, and the real data set here referred to Wine was found in University of

California, Irvine (UCI) Machine Learning Repository. As shown in table 1, Simdata contains 4050 samples and 07 classes while Wine contains 178 samples, 03 classes with 13 features each.

Table 1: Characteristics of used data sets

Data sets	Data size	Classes	Features	Dimension
Real data	178x13	03	13	02
Simulated data	4050x2	07	-	02

Source: The authors (2017)

3.2 Clustering Process

Spectral Clustering comprises three main successive steps namely: computing of similarity or affinity matrix between data points, update responsibilities and availabilities, use the clustering algorithm for identification of the *k* expected partitions, and then decide the exemplars (clusters). The flow chart summarizing the entire clustering process is shown in Figure 3.

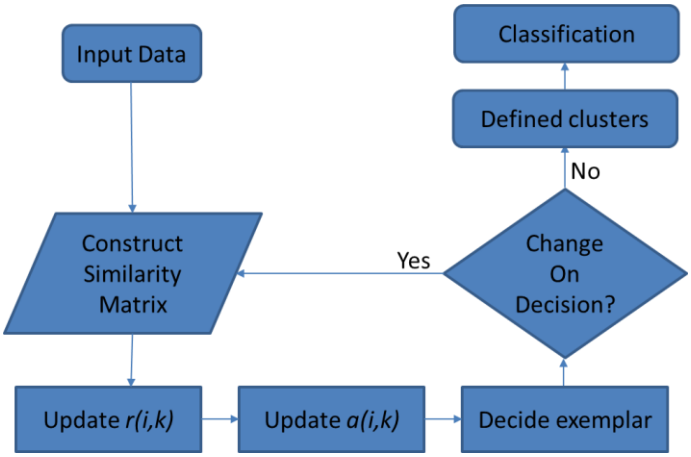

Figure 3: Flow chart summarising the methodology.

Considering an input data set with n points $C = \{C_1, C_2,...,C_N\}$ in R^2 to be categorized into k classes. The similarity matrix $S \in R^{N \times N}$ given by $S(i,j)$, which assumes values close to 0 when the data points C_i and C_j are dissimilar, and values close to 1 for points with similar features that can be grouped into the same cluster is constructed. Taking into account that each point $C_i \in C$ is transformed into $C_i^{'} \in R^k$ whose coordinates are: $\{Y_{im}, m \in \{1,...,k\}\}$, affinity propagation clustering is applied on this transformed set of points C' to calculate the evidence, and then assign each point C_i to the group at which its corresponding transformed point $C_i^{'}$ is assigned.

Evidence computation is the main task of affinity propagation clustering; it's where all the computation takes place. First, the parameters maximum iteration number, damping factor and preference need to be defined. The availabilities $a(i,k)$ are initialized to zero ($a(i,k) = 0$) for the first iteration, and then the message passing procedure is considered and the responsibilities $r(i,k)$ and availabilities $a(i,k)$ are updated from the similarity matrix according to equations (2) and (3) respectively.

The message propagation process stops as soon as it reaches a specified number of iterations or when the cluster structure stabilises with a given number of iterations. The exemplars are then decided based on the maximum similarity to find clusters, according to equation (4). Finally the exemplars are defined in a cluster and the process finished.

3.3 Clustering Accuracy Assessment

Given that AP variants output a series of clusters, it is necessary to use cluster validity indices to assess the quality of clustering results. According to Starczewski (2017), a given cluster, *Xj (j = 1,⋯, c),* the silhouette validity index *Sil* (*i = 1,..., n)* measures how excellent the sample is, indicating the *i*-th samples' membership in the cluster *Xj* and is given by the equation:

$$Sil(i) = \frac{mavd(i) - avd(i)}{\max\{avd(i), mavd(i)\}} \tag{5}$$

In the above equation, *avd(*i) represents the average distance measured from the *i*-th sample and all the samples in *Xj ; mavd(*i) is the averaged minimum distance between the *i*-th sample and all the samples clustered in X_k *(k =1, ... ,c) and* the maximum operator is given by "max". The index *Sil (i)* varies from -1 to 1. *Sil (i)* close to 1, could mean that the *i*-th sample has been accurately clustered and attributed to the suitable cluster. Values of *Sil (i)* tending to zero suggest that the sample could be attributed to the nearest cluster; and *Sil (i)* close to -1, could mean that there was a misclassification in such a sample (Rousseeuw 1987). Thus, for a given cluster, *Yj (j = 1,..., c),* it is possible to calculate a cluster Silhouette *Sijl,* which characterizes the heterogeneity and isolation properties of such a cluster:

$$S_{ij}l = \sum_{i=1}^{i=m} S_i l\,(i) \tag{6}$$

In the equation, *m* is the number of samples in $S_{ij}l$. For any partition $U \leftrightarrow Y : Y_l\ U \ldots Y_i\ U \ldots Y_c$, a Global Silhouette value, GS_u, can be used as an effective validity index for a partition *U*.

$$GS_u(i) = \frac{1}{c}\sum_{j=1}^{j=c} S_j \tag{7}$$

Furthermore, the equation (7) can be applied to estimate the most appropriate number of clusters for *U*. In this case the partition with the maximum GS_u is taken as the optimal partition.

4. Experimental Results

In the experiments, the elapsed times vs number of iterations was compared for AP, PAP and AAP, and the Silhouette validity index analysed. The results of this study are presented in the Tables 2, 3, 4 and Figures 4 and 5. The Table 2 presents the experimental results using simulated data while Tables 3 presents the results with real data sets.

Table 2: Performance of AP and its variants based on minimum and median preferences for simulated data

Algorithm	Minimum Preference			Median Preference		
	Clusters	Iterations	Time	Clusters	Iterations	Time
AP	05	4460	126.177	07	9634	376.912
PAP	05	4441	125.633	08	9509	372.036
AAP	04	8325	518.119	05	13098	717.000

According to the results using simulated data (Table 2), the minimum value of preference for AP algorithm resulted in 5 clusters within 376.912 seconds, while the median preference resulted in 7 clusters taking 126.177 seconds. PAP algorithm resulted in 5 clusters during 372.036 seconds for minimum preference values, and 8 clusters within 125.633 seconds for median values of preference. Compared to AP and PAP, AAP spent much time to find the exemplars. For minimum preferences, AAP found 4 clusters and 5 clusters when used median preference. The elapsed times were 717 and 518.119 seconds respectively for minimum and median preferences.

Table 3: Performance of AP and its variants based on minimum and median preferences for real data

Algorithm	Minimum Preference			Median Preference		
	Clusters	Iterations	Time	Clusters	Iterations	Time
AP	02	277	0.330	03	167	0.495
PAP	02	111	0.130	03	157	0.129
AAP	03	2423	3.938	03	2551	3.963

On the other hand, experiments with real data set (Table 3) gave the following results: Applying AP with minimum preference resulted in 2 clusters in the time period of 0.330 seconds. The median value of preference resulted in 3 clusters in 0.495 seconds. Similarly to AP, PAP algorithm also resulted in 2 and 3 clusters for minimum and median preferences and the elapsed times of 0.130s and 0.129s respectively. Differently from AP and PAP, AAP observed a considerable delay in finding the clusters. For both minimum and median preferences, AAP found 3 clusters within 3.938s and 3.963s respectively. The results also confirm that PAP performed faster than AP and AAP. This is due to the sub-matrix decomposition that optimised its performance as stated in the literature by Xia *et al*.(2008). Although AAP is the slowest algorithm as it tries to eliminate the oscillations and adjust the preference, it provides more stable solutions as can be shown by the Silhouette index from Table 4.

Table 4: Evaluation of AP variants based on Silhouette validity index

Data set	Algorithm	Clusters	Iterations	Time (s)	Silhouette
Simulated data (Simdata)	AP	07	810	144.520	0.367
	PAP	08	704	125.633	0.455
	AAP	04	13091	1592.340	0.660
Real data (Wine)	AP	11	749	28.520	0.465
	PAP	03	639	2.410	0.660
	AAP	02	2551	4.793	0.741

The Table 4 compares the Silhouette validity indices of the three algorithms studied. The results show that for both data sets, AAP performed well as its Silhouette indices were 0.741 and 0.660 much closer to one, which according to Rousseeuw (1987) means well separated clusters. AP performed badly for both data sets, having 0.465 and 0.367 respectively for real and simulated data, with the smallest indices of all. PAP was moderate with 0.660 when using real data and badly with simulated data, having 0.455.

The Figures 4 and 5 below present the visual clustering results for simulated and real data sets respectively, using median and minimum preferences. In the Figures, the colours represent different clusters obtained through the application of AP, PAP and AAP clustering algorithms. The number of clusters for both simulated and real data sets is the same as that presented in the tables 2 and 3. According to Figure 4, the clustering based on simulated data was not that accurate if compared to the clustering on real data sets. The Silhouette validity indices obtained from AP, PAP and AAP (0.367, 0.455 and 0.660) with simulated data (Table 4) were lower than those derived from real data sets, justifying the consistency of the real data in relation to the simulated.

Looking to the results in Figure 4 there are observations deviating markedly from other observations in the sample. These outliers may be due to random variation on simulated data or may indicate the need to undertake a careful prior examination of the dataset and to consider the use of robust statistical techniques such as fuzzy clustering. On the other hand, the results obtained based on real data sets (Figure 5), appear to be more accurate. The Silhouette indices (Table 4) from AP, PAP and AAP (0.465, 0.660 and 0.741) were greater than those obtained from simulated data and close to 1, what may infer that the sample has been well-clustered.

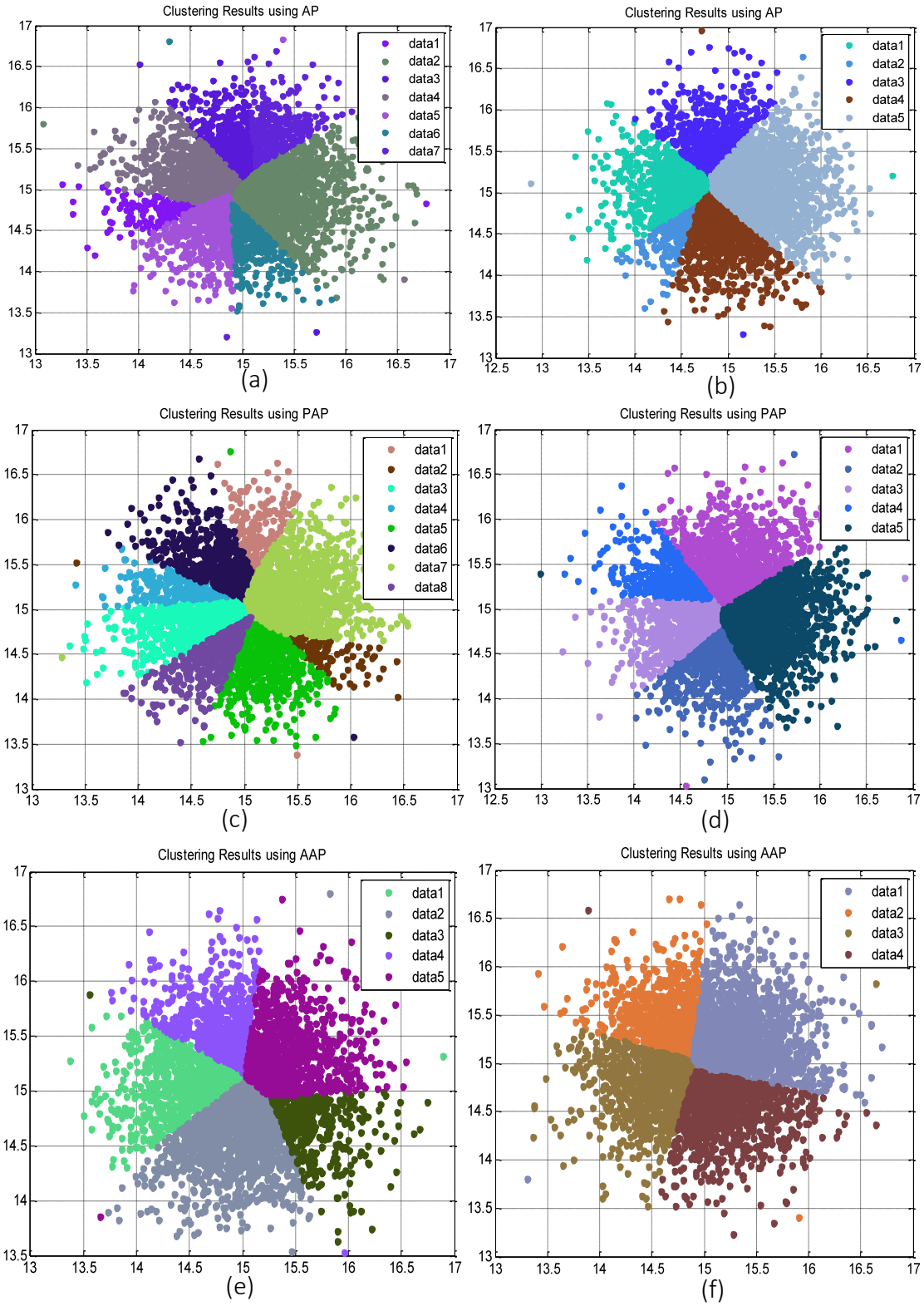

Figure 4: Clustering results for simulated data, considering the median and minimum preferences, respectively for AP in (a), (b), for PAP in (c), (d) and for AAP in (e) and (f).

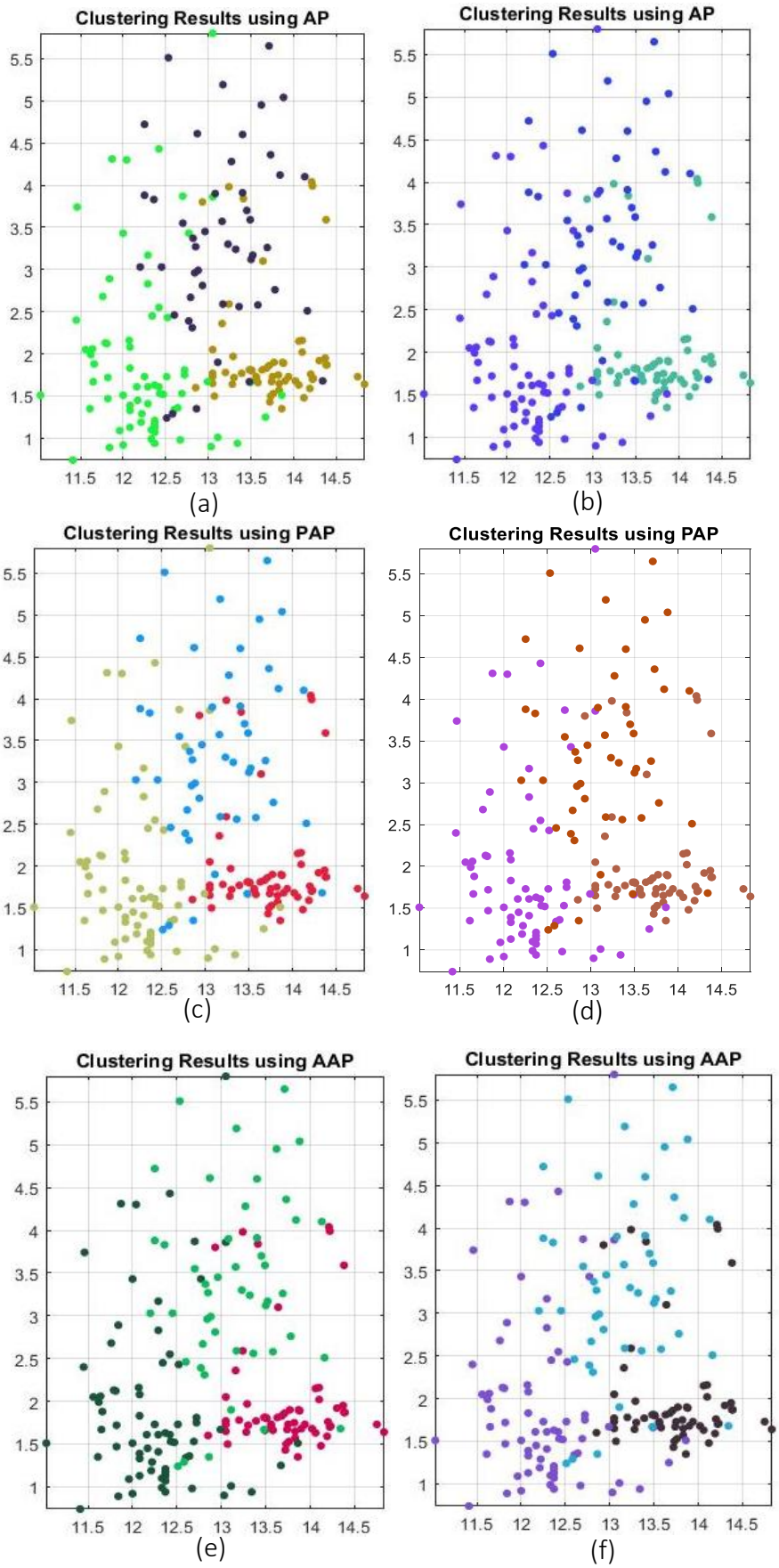

Figure 5: Clustering results for real data, considering the median and minimum preferences, respectively for AP in (a) and (b), for PAP in (c) and (d) and for AAP in (e) and (f).

5. Conclusions

The aim of this paper was to evaluate the clustering performance of AP algorithm, considering the influence of the preference parameter and damping factor. To undertake the study AP algorithm, PAP and AAP were used, and based on the discussed results the following conclusions were made: The clustering results obtained with simulated data, using minimum and median values of preference parameter for AP algorithm resulted respectively in 5 clusters within 376.912s and 7 clusters during 126.177s. PAP algorithm resulted in 5 clusters during 372.036s for minimum preference values, and 8 clusters within 125.633s for median values of preference. Compared to AP and PAP, AAP was the slowest algorithm in finding the exemplars. For minimum preferences, AAP found 4 clusters in 717 seconds, and took 518.119s to find 5 clusters when used median preference. According to Wang *et al*.(2008), this delay is due to the two-stage process of AAP aiming to eliminate the oscillations during the message passing mechanism and adjust the preference parameter.

Experiments for AP with real data sets resulted in 2 clusters within 0.330s using minimum preference. The median value of preference resulted in 3 clusters in 0.495s. Similarly to AP, PAP algorithm also found 2 and 3 clusters for minimum and median preferences and the elapsed times were 0.130s and 0.129s respectively. As in the case with simulated data, using real data sets AAP observed a considerable delay in finding the clusters. For both minimum and median preferences, AAP found 3 clusters within 3.938s and 3.963s respectively. The results also confirm that PAP performed faster than AP and AAP. This is due to the sub-matrix decomposition that optimised its performance as stated by Xia *et al*.(2008).

The result from Table 4 compares the Silhouette validity indices of the three algorithms studied. The results discussed show that for both data sets, AAP performed well as its Silhouette indices were 0.741 and 0.660 much closer to one, which according to Rousseeuw (1987), means well separated clusters. AP performed badly for both data sets, having 0.465 and 0.367 respectively for real and simulated data, with the smallest indices of all. PAP was moderate with 0.660 when using real data and badly with simulated data, having 0.455.

The Figures 4 and 5 shows that the number of clusters for both simulated and real data sets is the same as that presented in the Tables 2 and 3. According to Figure 4, the clustering based on simulated data was not that accurate if compared to that on real data sets. The Silhouette validity indices obtained from AP, PAP and AAP (0.367, 0.455 and 0.660) with simulated data (Table 4) were lower than those derived from real data sets. This justifies the consistency of the real data in relation to the simulated data. Looking to the results in Figure 4 there are observations deviating markedly from other observations in the sample. These outliers may be due to random variation on simulated data or may indicate the need to undertake a careful prior examination of the dataset and to consider the use of robust statistical techniques such as fuzzy clustering. On the other hand, the results obtained based on real data sets (Figure 5), appear to be more accurate. The Silhouette indices (Table 4) from AP, PAP and AAP (0.465, 0.660 and 0.741) were greater than those obtained from simulated data and close to 1, what may infer that the sample has been well-clustered.

In conclusion, the results show that these two parameters greatly influence the final clustering of AP algorithm. In terms of accuracy AAP performed well than the other two algorithms for both simulated and real data sets as the Silhouette index is close to one. The clustering accuracy obtained from AP, PAP and AAP when applying simulated data was low (Silhouette close to zero)

and presented outliers that may be attributed to the random variation of the data. Considering the speed efficiency, PAP was the fastest algorithm of all, taking the shortest time to perform the clustering process with reduced number of iterations.

References

Aggarwal, C.C. 2003. Towards systematic design of distance functions for data mining applications. In: *Proceeding of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* pp. 9-18, New York, USA, August.

Aggarwal, C.C. Reddy, C.K. 2011. *Data Clustering: Algorithms and Applications.* CRC Press.

Batra, A. 2011. Analysis and Approach: K-Means and K-Medoids Data Mining Algorithms. In: ICACCT, *5th IEEE International Conference on Advanced Computing & Communication Technologies*. ISBN 81-87885-03-3 pp. 274-279.

Bombatkar, A. Parvat, T. 2015. Improvements In Clustering Using Affinity Propagation: A Review. *Journal of Multidisciplinary Engineering Science and Technology,* 2(6), pp. 3159-0040.

Cattell, R. B. 1943. The description of personality: Basic traits resolved into clusters. *Journal of Abnormal and Social Psychology*, 38, pp. 476-506.

Chehdi, K. Soltani, M. Cariou, C. 2014. Pixel classification of large size hyperspectral images by affinity propagation. *Journal of applied remote sensing*, 8(1), pp. 1-14.

Dermoudy, J. Kang, Byeong-Ho. Bhattacharyya, D. Jeon, Seung-Hwan. Farkhod, A.A. 2009. Process of Extracting Uncover Patterns from Data: A Review. *International Journal of Database Theory and Application*, 2(2).

Dey, V. Zhang, Y. Zhong, M. 2010. A Review on image segmentation techniques with remote sensing perspective**.** In: *IAPRS.* Vienna, Austria, 5-7 July.

Driver, H.E. Kroeber, A.L. 1932. Quantitative expression of cultural relationships. *American Archaeology and Ethnology*, 31, pp. 211-56.

Duda, R.O. Hart, P.E. 1973. *Pattern Classification and Scene Analysis.* Vol. 3, Wiley, NewYork, USA.

Dueck, D. 2009. Affinity Propagation: clustering data by passing messages. *University of Toronto*, 24 September, Toronto, Canada.

Fanhua, S. Jiao, L.C. Jiarong, S. Fei, W. Gong, M. 2012. Fast affinity propagation clustering: A multi-level approach. *Pattern Recognition,* pp. 474-486.

Frey, B.J. Dueck, D. 2007. Clustering by passing messages between data points. *Science*, 315(5814), pp. 972-976.

Galdi, P. Napolitano, F. Tagliafe, R. 2014. A comparison between Affinity Propagation and assessment based methods in finding the best number of clusters. In: *Proceedings of CIBB*.

Givoni, I.E. Frey, B.J. 2009. A Binary Variable Model for Affinity Propagation. *Journal of Neural Computation*, 21(6), pp. 1589-1600.

Guan, R. Shi, X. Marchese, M. Yang, C. Liang, Y. 2011. Text clustering with seeds affinity propagation. *IEEE Transactions on Knowledge and Data Engineering*, 23(4), pp. 627-637.

Jain, A.K. Dubes, R.C. 1988. *Algorithms for Clustering Data.* Prentice Hall, Upper Saddle River, NJ, USA.

Jia, Y. Wangz, J. Zhangy, C. Hua, X.S. 2008. Finding Image Exemplars Using Fast Sparse Affinity Propagation. In: *Proceedings of the 16th ACM International conference on Multimedia*.

Plangprasopchok, A. Lerman, K. Getoor, L. 2010. Integrating structured metadata with relational affinity propagation. In: *Proceedings in Statistical Relational Artificial Intelligence*.

Refianti, R. Mutiara, A.B. 2016. Performance Evaluation of Affinity Propagation Approaches on Data Clustering. *(IJACSA) International Journal of Advanced Computer Science and Applications,* 7(3).

Rousseeuw, P.J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal. Comp App. Math,* 20, pp. 53-65.

Serdah, A.M. Ashour, W.M. 2016. Clustering large-scale data based on modified affinity propagation algorithm. *JAISCR*, 6(1), pp. 23-33.

Soman, C. Ali, A. 2015. Incremental Affinity Propagation Clustering with Feature Selection. *International Journal of Innovative Research in Computer and Communication Engineering*, 3(10).

Starczewski, A. A new validity index for crisp clusters. *Pattern Analysis and Applications*, 20(3), pp. 687-700.

Sun, C. Wang, C. Song, S. Wang, Y. 2009. A Local Approach of Adaptive Affinity Propagation Clustering for Large Scale Data. In: *Proceedings of International Joint Conference on Neural Networks.* Atlanta, Georgia, USA, 14-19 June.

Thavikulwat, P. 2014. Affinity propagation: a clustering algorithm for computer-assisted business simulations and experiential exercises. *Developments in Business Simulation and Experiential Learning*, 35.

Tryon, R.C. (1939). *Cluster Analysis: Correlation Profile and Orthometric (factor) Analysis for the Isolation of Unities in Mind and Personality*. Edwards Brothers.

Wang, K. Zhang, J. Li, D. Xinna-Zhang, T. Guo, T. 2008. Adaptive Affinity Propagation Clustering. *Acta Automatica Sinica*, 33(12), pp. 1242-1246.

Xia, D.Y. Wu, F. Zhang, X.Q. Zhuang, Y.T. 2008. Local and global approaches of affinity propagation clustering for large scale data. *Journal of Zhejiang University Science A*, 9(10), pp. 1373-1381.

Yang, C. Bruzzone, L. Sun, F. Lu, L. Guan, R. Liang, Y. 2010. A fuzzy statistics-based affinity propagation technique for clustering in multispectral images. *IEEE Transactions in Geoscience and Remote Sensing,* 48(6), pp. 2647-2659.

Zhang, K. Gu, X. 2014. An Affinity Propagation Clustering Algorithm for Mixed Numeric and Categorical Datasets. *Mathematical Problems in Engineering*, 2014, pp. 1-8.

Zhang, X. Wu, F. Xia, D. Zhuang, Y. 2008. *Partition Affinity Propagation for Clustering Large Scale of Data in Digital Library*. College of Computer Science, Zhejiang University, Hangzhou, China.

Zhou, Y. Sun, G. Xing, Y. Wang, Y. 2015. Community Detection Algorithm Based on Adaptive Affinity Propagation. *Journal of Computational Information Systems,* 11(22), pp. 8101-8110.