A swarm-inspired projection algorithm

Mu-Chun Su\textsuperscript{a,∗}, Shi-Yong Su\textsuperscript{a}, Yu-Xiang Zhao\textsuperscript{b}

\textsuperscript{a}Department of Computer Science and Information Engineering, National Central University, Taiwan, ROC
\textsuperscript{b}Department of Computer Science and Information Engineering, Ta Hwa Institute of Technology, Taiwan, ROC

\textbf{Article Info}

\textbf{ABSTRACT}

In this paper we propose a new data projection algorithm.\footnote{Corresponding author. \hspace{1em} E-mail address: muchun@csie.ncu.edu.tw (M.-C. Su).} We name the new data projection the swarm-inspired projection (SIP) algorithm. The algorithm allows us to visually estimate the number of clusters existing in a data set. Based on the projection result, we may then partition the data set into the corresponding number of clusters. The SIP algorithm regards each data pattern in a data set as a piece of crumb which will be sequentially tossed to a flock of doves on the ground. The doves will adjust their physical positions to compete for crumbs. Gradually, the flock of doves will be divided into several groups according to the distributions of the crumbs. The formed groups will naturally correspond to the underlying data structures in the data set. By viewing the scatter plot of the final positions of the doves we can estimate the number of clusters existing in the data set. Several data sets were used to demonstrate the effectiveness of the proposed SIP algorithm.

\section{Introduction}

Recently, many properties of social insect (or animals) collective behaviors have attracted a great amount of attention from researchers\cite{1,2}. Social insects (or animals) provide us with a powerful concept to create decentralized systems of simple interacting, and often mobile, agents (e.g., ants, bees, birds). A rich source of mechanisms in social insect collective behaviors may serve as metaphors for designing the so-called swarm-intelligence-based systems. Swarm intelligence, a form of artificial intelligence, is the emergent collective intelligence of groups of simple agents. Different swarm-intelligence-based systems are inspired by different subsets of the available metaphors. Ant colony systems (ACS)\cite{3,4} and particle swarm optimization (PSO)\cite{5,6} are two well-known kinds of swarm intelligence. The applications of these systems are widely spread from optimization, communications networks, to robots\cite{7–11}.

Cluster analysis is one of the basic tools for exploring the underlying structure of a given data set and is being applied in a wide variety of engineering and scientific disciplines such as medicine, psychology, biology, society, pattern recognition, and image processing. The primary objective of cluster analysis is to partition a given data set into so-called homogeneous clusters such that patterns within a cluster are more similar to each other than patterns belonging to different clusters. There are two major difficulties encountered in clustering data: (1) cluster geometric shapes are full of variability and (2) the number of clusters is not always known a priori. Different distance measures lead to different types of clusters (e.g., compact hyper-spheres, compact hyper-ellipsoids, lines, shells, etc.). Recently, several clustering algorithms with different distance measures have been developed for clustering data sets with different geometric shapes\cite{12–22}. These algorithms are used to detect compact clusters\cite{12–15}, straight lines\cite{14–16}, shells\cite{17–19}, contours with polygonal boundaries\cite{19,20}, or clusters of different geometrical structures\cite{21,22}.

In fact, if cluster analysis is to make a significant contribution to engineering applications, much more attention must be paid to the determination of the optimal number of clusters. Basically, there are four different approaches to the determination of the cluster numbers. The first approach is to use some global validity measure to validate clustering results for a range of cluster numbers. Generally, the determination of the optimum cluster number using global validity measures is very expensive since clustering has to be carried out for a variety of possible cluster numbers. Therefore, the idea of performing progressive clustering was proposed\cite{23–26}. Since the optimal partition cannot be found in a single run, one could filter the presumably good clusters, eliminate spurious clusters, and merge compatible clusters into larger clusters. This process is continued until the final partition contains good clusters only. The third approach is to perform competitive clustering\cite{27,28}. By the mechanism of competing for data points only good clusters can survive. Another approach is the use of the family of projection algorithms. Projection algorithms can project high-dimensional data onto a low-dimensional space to facilitate visual inspection of the data. They allow us to visualize high-dimensional data as a two-dimensional
The swarm-inspired projection algorithm which was inspired by part of the mechanisms in the doves’ foraging behavior. One thing should be emphasized is that the focus of the paper is not to precisely model the doves’ foraging behavior, but to show that some basic foraging principles may be useful for data projection.

2.1. Motivations

Imagine the following picture. On a sunny day, plenty of doves are foraging at a plaza. Suddenly, a child is tossing crumbs for the doves. Understandably, these doves will pace up and down for a while and then quickly encircle the child to compete for the crumbs. Suppose another child suddenly appears at a different location of the plaza and also tosses crumbs for the doves. One may observe that some doves which fail in the competition of the crumbs tossed by the first child may start to move toward the second child. Gradually, two groups of doves will be formed and, respectively, encircle the corresponding child, as shown in Fig. 1.

The doves’ foraging behavior motivated us to propose a new data projection algorithm. In this method, each data pattern, \( x \), in a data set is regarded as a piece of artificial crumb and these artificial crumbs (i.e., data patterns) will be sequentially tossed to a flock of doves on a two-dimensional artificial ground. The flock of doves adjusts its physical movements to seek these artificial crumbs. Individual members of the flock can profit from discoveries of all of the other members of the flock during the foraging procedure because an individual is usually influenced by the success of the best individual of the flock and thus has a desire to imitate the behavior of the best individual. Gradually, the flock of the doves will be divided into several groups according to the distributions of the artificial crumbs. Those formed groups will naturally correspond to the underlying data structure in the data set. By viewing the distributions of the doves on the two-dimensional artificial ground one may quickly estimate the number of clusters inherent in the data set.

The aforementioned idea can work well for 2-dimensional cases; however, many practical data sets have high-dimensional data points. Therefore, how to generalize the aforementioned idea to process high-dimensional data is a very demanding challenge. In the real world, each dove has a pair of eyes to find out where crumbs are, but in the artificial world, how can a dove

![Fig. 1. Two groups of doves are formed when two different groups of children toss crumbs for the doves.](image)
perceive a piece of multi-dimensional artificial crumb located around it?

When we were perplexed about how to make a dove be able to perceive multi-dimensional artificial crumbs, a thought suddenly occurred to us that we could borrow some ideas from our previous work, the DSOM algorithm [33] which extends the capability of the conventional self-organizing feature map algorithm to allow a user to visualize high-dimensional data as a two-dimensional scatter plot. In the DSOM algorithm, each neuron is associated with a multi-dimensional weight vector, \( \mathbf{w}_j \), and a two-dimensional position vector, \( \mathbf{p}_j \). The vector \( \mathbf{p}_j \) determines the position of neuron \( j \) in the network structure. By the DSOM algorithm, the network will adaptively adjust its network structure during the learning phase so as to make neurons responding to similar stimulus have similar weight vectors and spatially move nearer to each other at the same time. The resulting representations allow a qualitative analysis of the inherent structure of clusters within the input data. The success of the DSOM algorithm is critically dependent on how the values of its main parameters are determined; therefore, the stability of the DSOM algorithm sometimes is not that high.

Similar to the idea in the DSOM algorithm, each dove is with a multi-dimensional artificial sense organ represented as a sense organ vector, \( \mathbf{w} \), which has the same dimensionality as a data pattern, \( \mathbf{x} \), and a two-dimensional position vector, \( \mathbf{p} \), which represents its position on the two-dimensional artificial ground. In addition to the two vectors, \( \mathbf{w} \) and \( \mathbf{p} \), a parameter called the satiety parameter is attached to each dove. This special parameter endows a dove with the ability of expressing its present satiety status with respect to the food. If a dove is with a low degree of satiety then it would have a strong desire to change its present foraging policy and be more willing to imitate the behavior of the dove which performs the best among the flock.

Fig. 2. The movements of 49 doves on the two-dimensional artificial ground: (a) the initial positions of the 49 doves; (b) the positions of the 49 doves at the end of 5 epochs; (c) the positions of the 49 doves at the end of 10 epochs; and (d) the positions of the 49 doves at the end of 15 epochs.
Fig. 3. The five data sets. For the high-dimensional data sets, data points are projected to two-dimensional space by the Sammon's algorithm: (a) the two-ellipse data set; (b) the chromosome data set; (c) the four-dimensional iris data set; (d) the nine-dimensional breast-cancer data set; and (e) the 20-dimensional non-overlapping data set.
We sort a list of symbols to ease the understanding of the description of the SIP algorithm.

- $\mathbf{x}$ denotes a data pattern in the data set
- $\mathbf{w}$ denotes the multi-dimensional sense organ vector assigned to a dove
- $\mathbf{p}$ denotes the two-dimensional position vector associated with a dove
- $e$ denotes the number of epochs
- $k$ denotes the time index
- $f'_j$ denotes the degree of satiety of the $j$th dove at the $e$th epoch
- $b_f$ denotes the dove nearest to the food
- $b_s$ denotes the dove with the highest degree of satiety
- $M \times N$ denotes the number of doves

2.2. Training phase

Step 1: Decide the number of doves and then deploy them on the two-dimensional artificial ground. Assume that the number of doves is pre-specified to be $M \times N$. These doves can be randomly distributed on the artificial ground; however, we suggest deploying them uniformly on a rectangular region (i.e., $p_1 = (0, 0)^T$, $..., p_{M \times N} = (M-1, N-1)^T$), as shown in Fig. 2(a).

Step 2: Set the number of epochs, $e = 0$ and set the degree of satiety, $f'_j = 0$ for $j = 1, ..., M \times N$. Initialize the multi-dimensional sense organ vector, $\mathbf{w}_j$, for $j = 1, ..., M \times N$. The initialization of the sense organ vector $\mathbf{w}_j$’s can be done in three ways. The simplest way is to randomly initialize the sense organ vectors to be small valued vectors. The second way is to randomly select the same
number of data patterns from the data set and randomly assign them to be the doves’ initial sense organ vectors. The third way is to follow a basic intuition. In the real world, doves which are close to each other will have similar vision. Following the same idea, in the artificial world, neighboring doves should have similar multi-dimensional sense organ vectors. By adopting the special initialization method [52], doves with similar position vectors will be assigned to similar multi-dimensional sense organ vectors. The basic idea of the special initialization method [52] is to squeeze the $n$-dimensional hyper-rectangle into a two-dimensional plane so a two-dimensional nest can effectively cover the plane. Starting from this kind of initialization, the training procedure can be accelerated. In addition, since the basic idea of the special initialization method is to squeeze the $n$-dimensional hyper-rectangle into a two-dimensional plane, doves can be deployed to effectively cover the $n$-dimensional data space without the risk of converging to one small area.

Step 3: Compute the total amount of the degrees of satiety in the flock, $T(e) = \sum_{j=1}^{M \times N} I_j$.

Step 4: Present a piece of artificial crumb (i.e., an input pattern) $x_k$ to the $M \times N$ doves.

Step 5: Locate the dove $b_f$ nearest to the crumb $x_k$ using the minimum-distance criterion:

$$ b_f = \arg \min_j \|x_k - w_j(k)\|, \quad \text{for } j = 1, \ldots, M \times N \quad (1) $$

The dove with the artificial sense organ vector which is most similar to the artificial crumb, $x_k$, is claimed to be the winner.
Step 6: Update each dove’s satiety degree by using the following equation:

\[ f_j^{(\text{new})} = \frac{\|x_k - w_j^b(k)\|}{\|x_k - w_j(k)\|} + \lambda f_j^{(\text{old})}, \quad \text{for} \quad j = 1, \ldots, M \times N \]  

(2)

The first term on the right-hand-side of (2) indicates that if a dove has a sense organ vector similar to the dove \( b \)'s sense organ vector (i.e., \( \|w_j^b(k) - x_k\| = \|w_j(k) - x_k\| \)) then it can achieve a higher sense of satiety with respect to the present crumb than other doves since it will probably have the chance of sharing the present crumb. The second term on the right-hand-side of (2) indicates that the degree of satiety will decay as time increases. The satiety decay is implemented by introducing a coefficient of decay \( \lambda, 0 \leq \lambda < 1 \).

Step 7: Select the dove, \( b_i \), with the highest degree of satiety using the following maximum criterion:

\[ b_i = \arg \max_{1 \leq j \leq M \times N} f_j^c \]  

(3)

The dove, \( b_i \), selected by (3) is the dove which presents the best foraging performance and it is the one which deserves to be imitated by other doves in the flock.

Step 8: Update the sense organ vectors and the position vectors according to the following equations:

\[ w_j(k+1) = \begin{cases} w_j^b(k) + \eta \|x_k - w_j^b(k)\| & \text{for} \ j = b_f \\ w_j(k) & \text{for} \ j \neq b_f \end{cases} \]  

(4)

\[ p_j(k+1) = p_j(k) + \eta_b \beta (p_j^b(k) - p_j(k)) \quad \text{for} \ j = 1, \ldots, M \times N \]  

(5)

Fig. 6. The projection performance for the iris data set: (a) the LF algorithm; (b) the DSOM algorithm; and (c) the SIP algorithm.
where
\[ \beta = \left( \frac{\text{f}_{\text{bs}} - \text{f}_{\text{bs}}^{(\text{new})}}{\text{f}_{\text{bs}}^{(\text{new})}} \right) \frac{\| \text{w}_{\text{bs}} - \text{w}_{\text{bs}}^{(k)} \|}{\| \text{w}_{\text{bs}}^{(k)} \|} \left( 1 - \frac{\| \text{p}_{\text{bs}}^{(k)} - \text{p}_{\text{bs}}^{(k)} \|}{L} \right) \]
(6)

\[ L = \sqrt{M^2 + N^2} \]
(7)

The parameters, \( \eta_{\text{ws}} \) and \( \eta_{\text{ps}} \), are the learning rates for updating the sense organ vector and the position vector, respectively. Detailed descriptions of the updating rules (4)–(6) will be given in Section 2.3.

Step 9: Go to step 4 until all data patterns are processed.

Step 10: Terminate the whole training procedure if the following criterion is met:
\[ \sum_{j=1}^{N \times M} \left| f_j^{(\text{new})} - T(e) \right| \leq \varepsilon \]
(8)

Otherwise, increase the number of epochs by one (i.e., \( e = e + 1 \)) and go to step 3 until the number of epochs reaches a pre-specified limit. If the criterion given in (8) is satisfied then it means that the total amount of satiety converges to some extent.

The SIP algorithm has the order of complexity, \( O(MNNde) \) where \( N_d \) is the number of data points in the data set, \( M \times N \) is the number of doves, and \( e \) is the number of epoch.

2.3. Physical interpretations of updating rules

For easier understanding, we interpret the updating rules given in (4)–(6) in the following ways:

1. Having won a competition the winner would further exploit its skills to win again if a similar situation is encountered. The
Fig. 8. The visualization effect comparison for the 20-dimensional non-overlapping data set: (a) the LF algorithm; (b) the DSOM algorithm; and (c) the SIP algorithm.

The updating rule given in (4) has the overall effect of moving the winner’s sense organ vectors $w_j$ toward the input pattern $x_k$. That is, the sense organ vectors, $w_j$, are expected to become prototypes of the input vectors. Note that only the dove $b_j$'s sense organ vector is updated. This updating rule is different to the updating rule in the conventional SOM algorithm and the DSOM algorithm.

(2) An individual is influenced by the success of the best individual in the flock and tries to imitate the behavior of the best individual [1]. That is, doves will move toward the dove with the highest degree of satiety to find more food. This kind of social learning is simulated by updating the position vector $p_j$ to be more like the position vector of the dove with the highest degree of satiety $p_{bs}$ (i.e., $p_j(k + 1) = p_j(k) + \eta_p[p_{bs}(k) - p_j(k)]$).

(3) When an individual is with a higher degree of satiety it is prone to become conservative and would hesitate to change its present foraging policy. On the contrary, when an individual is with a lower degree of satiety it would probably have a strong desire to change its present foraging policy and be more willing to imitate the behavior of the best individual. This kind of social influence is simulated by making the adjustment proportional to the value of the first term on the right-hand-side of (6) (i.e., $(f_{bs} - f_j)/f_{bs}^2$).

(4) The nearer an individual is to the newly tossed crumb the stronger desire it would have to compete for food. Therefore, we make the adjustment amount proportional to the value of the second term on the right-hand-side of (6) (i.e., $\|x_k - w_{bf}(k)\|/\|x_k - w_j(k)\|$).

(5) Basically, social influence will gradually decay as it spreads out [1]; therefore, the degree of influence is inversely proportional to the distance between the individual and the best individual in the flock. This kind of social influence is simulated by making the adjustment amount proportional to the value of the third term on the right-hand-side of (6) (i.e., $1 - \|p_j(k) - p_{bs}(k)\|/L$).

2.4. Estimation of the cluster number

As mentioned in the introduction section, there are four different approaches to the determination of the cluster numbers. Similar to the projection-based method, our method relies on human vision. By visualizing the scatter plots of the doves' position vectors, $p_j$'s, we can explore various properties, such as the clustering tendency and the underlying structure of the data set. Since the comprehensive effect of the updating rules given in (4) and (5) will force doves to move to their respective nearest food sources, one may expect that doves will gradually be partitioned into several groups which are correspondent with the underlying structure of the data set. Once the final locations of the position vectors of the doves have been displayed on an image or a sequence of scatter plots has been displayed in a movie-like manner, one can visually determine the number of clusters. Each food source (i.e., data cluster) will, respectively,
will have a high tendency towards presenting overlapped groups. Then the analyst will be placed in a difficult position and different analysts may select a different number of clusters based on one scatter plot. After we have successfully estimated the number of clusters by visual examination of the scatter plot, we enter the second stage of the clustering phase.

3. Relations to other algorithms

In this section, we explore the similarities and differences between the proposed SIP algorithm and some existing swarm-based algorithms. As mentioned in the introduction section, several swarm-intelligence-based data clustering algorithms have been proposed in recent years [35–49]. Each approach has its own applicability, advantages, and limitations.

The basic idea of the PSO-based approaches [35–39] is the use of the PSO algorithm as an optimization tool for optimizing some kind of cluster validity measure, such as the quantization error, the inter-distance between any pair of clusters, or the intra-distance between data points and their cluster means. Their advantage is that they may avoid the local minimization problem incurred from the use of gradient-based optimization tools. However, one crucial problem encountered by them is that they still need to pre-specify the number of clusters before they can proceed to the optimization procedure. A PSO-based clustering approach (DCPSO) was newly introduced to automatically determine the optimum number of clusters and simultaneously cluster the data set [40]. The algorithm starts by partitioning the data set into a relatively large number of clusters and then uses binary PSO to select the best number of clusters based on the computation of some cluster validity measure. Finally, the centers of the chosen clusters are refined via the k-means algorithm.

Unlike these PSO-based approaches, the ant-based algorithms [41–45], the information-flocking-based algorithms [46–49] and our proposed SIP algorithm do not optimize some kind of cluster validity measure. Although these three classes of algorithms were all bio-inspired, they were motivated by different metaphors of social animals’ behaviors. The ant-based algorithms were motivated by the clustering and sorting behavior of ants. The general idea of ant-based algorithms is that isolated items will be picked up by artificial ants and then dropped at some other location where more items of that type are present. The ant-based algorithms could generate some interesting clustering results only if their parameters can be appropriately selected. In addition, Bonabeau et al. have pointed out that the LF algorithm [42], one of the popular ant-based algorithms for data clustering, does not appear to be very efficient in terms of computation time. As for the information-flocking-based algorithms [46–49], they allow the user to visualize data based on the schooling behavior of animals such as fish, birds. The information flocking concept was originally proposed by Proctor and Winter [46]. In their simulations, each fish or individual “boid” represents a data point in the data set to be analyzed and these fish will swim along a heading vector according to four behavioral rules such as collision avoidance, velocity matching, origin homing, and information flocking. The calculation of the heading vector for a fish $A$ is as follows [46]:

\[
\text{New heading} = w_{\text{CA}} \sum -(b_i - A) + w_{\text{VM}} \sum (C - A) \\
+ w_{\text{OH}} \sum (A - O) + w_{\text{IF}} \sum S_{ij}(b_i - A) \tag{9}
\]

where $w_{\text{CA}}, w_{\text{VM}}, w_{\text{OH}}, w_{\text{IF}}$ is the weighting applied to Collision Avoidance, Velocity Matching, Origin Homing and Information Flocking behavior, respectively; $A$, $b_i$, $O$ the position vector of fish $A$, fish $i$ and origin, respectively, $C$ the position vector of the center of fish $A$'s neighbors, $S_{ij}$ the similarity of fish $A$ and fish $i$.  

Fig. 8. (continued).
Based on the aforementioned discussions, several observations can be concluded as follows:

1. In the artificial world of the information-flocking-based algorithms, each artificial agent represents a data point of a data set; therefore, the amount of artificial agents is equal to the amount of data points in the data set. But, in the two-dimensional artificial world of the proposed SIP algorithm, the amount of doves is less than the amount of data points. In addition, the movements of artificial agents in these two classes of algorithms are governed by different behavioral rules.

2. In the ant-based algorithms, artificial ants move in a discrete space; however, in the SIP algorithm, doves move in a real space instead of a grid. Of course, the movements of artificial agents in these two classes of algorithms are also governed by different behavioral rules.

3. All these algorithms provide a visualization of data to allow the user to explore the underlying structure of a data set.

As mentioned in Section 2.1, the SIP algorithm borrowed some ideas and notations from the DSOM algorithm; therefore, to some extent, the proposed SIP algorithm can be regarded as an improved version of the DSOM algorithm or a new interpretation of the DSOM algorithm. However, there are several differences between the DSOM algorithm and the proposed SIP algorithm.

1. These two algorithms were motivated from different metaphors. While the SIP algorithm was motivated by the metaphor of the doves' foraging behavior, the DSOM algorithm was developed to extend the capability of the conventional SOM algorithm which was motivated by a distinct feature of the human brain.
Fig. 10. The projection results achieved by setting $\eta_w = 0.1$, $\eta_p = 0.1$ and (a) $\lambda = 0.01$; (b) $\lambda = 0.05$; (c) $\lambda = 0.1$; (d) $\lambda = 0.3$; (e) $\lambda = 0.5$; and (f) $\lambda = 0.7$. 
(2) In addition to the vectors, $w$ and $p$, a new parameter referred to as the satiety parameter is introduced in the SIP algorithm to endow each dove with the ability of expressing its satiety status. Due to the inclusion of the satiety parameter, one may find that the DSOM algorithm and the SIP algorithm have different updating rules for the two vectors, $w$ and $p$. In the DSOM algorithm, the weight vectors of the winner and its neighbors will be updated together but in the SIP algorithm, only the dove, $b_f$, has the chance of updating its artificial sense organ vector.

(3) While there is only one stage for selecting a winner in the DSOM algorithm there are two stages for selecting winners, $b_f$ and $b_s$, in the SIP algorithm.

(4) For many cases, the SIP algorithm presents a better visualization effect than the DSOM algorithm. In addition, the SIP algorithm requires a smaller number of epochs than the DSOM algorithm does.

4. Simulation results

To test the performance of the SIP algorithm, five data sets consisting of two artificial data set and three real data sets. The projection capability of the SIP algorithm was compared with the Sammon's projection algorithm, the LF algorithm, and the DSOM algorithm.

For the SIP algorithm, the maximum number of epochs for every data set except the iris data set and the 20-dimensional data set were set to be 5. The maximum numbers of epochs for the iris data set and the 20-dimensional data set were set to be 10 and 20, respectively.
The maximum number of epochs for the LF algorithm and the DSOM algorithm were set to be 20,000, and 100, respectively.

4.1. Example 1: The two-ellipse data set

The first data set consists of two ellipses as shown in Fig. 3(a) and it contains 400 two-dimensional data points. Each ellipse is consisted of 200 data points and there is a clear border between these two clusters. Since the data set is consisted of two-dimensional patterns, we do not need to use the Sammon’s projection algorithm to process it. The best performance of the visualization effect achieved by the LF algorithm, the DSOM algorithm, and the SIP algorithm from 10 runs are shown in Fig. 4(a)–(c), respectively. These results were achieved by adopting 7×7 artificial agents (e.g., ants, neurons, and doves) in each algorithm with proper parameter values. By viewing Fig. 4(a) one may observe that the LF algorithm presents two large groups which are consisted of two small groups. However, the two large groups are not consisted of data points with the same class labels. As for the DSOM algorithm, Fig. 4(b) presents three loose groups; however, one may think that there are one small group on the left and one large group which is consisted of two small well-separated groups on the right. By viewing Fig. 4(c), one may find that there are one compact dense group and one line-shaped group. Based on the visualization result shown in Fig. 4(c), the number of clusters in the data set was estimated to be two. This example serves to illustrate that the SIP algorithm is able to detect two well separated ellipse-shaped clusters.

4.2. Example 2: The chromosomes data set

This example is used to demonstrate that the SIP algorithm is able to cluster well-separated clusters with arbitrary shapes. This image shown in Fig. 3(b) contains 11 chromosomes. We used an edge detector to convert the image into a binary image and then we
transformed the object pixels in the edge image to be data points. The best performance of the projection performance achieved by the LF algorithm, the DSOM algorithm, and the SIP algorithm from 10 runs are shown in Fig. 5(a)–(c), respectively. These results were achieved by adopting 7×7 artificial agents in each algorithm with proper parameter values. By viewing Fig. 7, we find that only the proposed SIP algorithm presents 11 compact groups. The number of groups matches the number of chromosomes in the data set. This example shows that the SIP algorithm could also work well with clusters of arbitrary shapes.

4.3. Example 3: The iris data set

This example is used to illustrate that the SIP algorithm may not work well as we expect if a data set consists of non-separable clusters. The iris data set has three subsets (i.e., iris setosa, iris versicolor, and iris virginica), two of which are overlapping. The iris data are in a four-dimensional space and there are a total of 150 patterns in the set. There are 50 patterns from each of the three species. We used the Sammon’s projection algorithm [29] to project the iris data set on the two-dimensional space as shown in Fig. 3(c). The best performance of the projection performance achieved by the LF algorithm, the DSOM algorithm, and the SIP algorithm from 10 runs are shown in Fig. 6(a)–(c), respectively. These results were achieved by adopting 7×7 artificial agents in each algorithm with proper parameter values. In this example, both the proposed SIP algorithm and the DSOM algorithm present two groups but the LF algorithm presents two large groups and one small group located at the upper right corner of the scatter plot shown in Fig. 6(a).

It is apparent that there are only two well-separated groups in Fig. 6(b) and (c). This observation is consistent with our a priori knowledge about the iris data set. It complies with the fact that one class of the iris data set can be linearly separated from the other.
two classes, and the two remaining classes are not separable. By further examining Fig. 6(c), one may notice that the doves corresponding to the two non-separable classes are not located at random positions but to some extent group together according to their similarities.

4.4. Example 4: The breast cancer data set

In this example, the Wisconsin breast cancer data set taken from the repository of the University of California at Irvine, Repository of Machine Learning Databases (UCI) [53] is used to illustrate that the SIP algorithm could be utilized to cluster a real life complicated data set. We briefly describe the data set. The problem is to predict whether a tissue sample taken from a patient’s breast is malignant or benign. There are two classes, nine numerical attributes, and 699 observations. Sixteen instances are removed because they contain a missing attribute value. We use the Sammon’s projection algorithm [29] to project the breast cancer data set onto two-dimensional space as shown in Fig. 4(d).

The best performance of the projection performance achieved by the LF algorithm, the DSOM algorithm, and the SIP algorithm from 10 runs are shown in Fig. 7(a)–(c), respectively. These results were achieved by adopting 7×7 artificial agents in each algorithm with proper parameter values. The scatter plot shown in the Fig. 7(a) does not present two-group structure at all. By carefully examining Fig. 7(b) and (c), one may notice that the doves corresponding to the two non-separable classes are not located at random positions but to some extent group together according to their similarities. The DSOM algorithm and the SIP algorithm barely present two clusters having one dense line-shaped group and one loose large group. Of course, a different user may make a different observation about the clustering tendency existing in Fig. 7(b) and (c). This example once again illustrate that the limitation of the SIP algorithm. That is, if data points are not well-separated then different analyzer
may have different estimates from viewing the same visualization results.

4.5. Example 5: The 20-dimensional non-overlapping data set

This artificial data set consists of 10 non-overlapping clusters and each cluster contains 100 20-dimensional data points. The data points from each cluster are generated as follows. For the ith cluster, the mean value and the standard deviation for the ith and the (i+1)th variables are 5 and 0.5, respectively. These two variables are all Gaussian-distributed. As for the remaining variables, they are all zeros. The Fig. 4(e) shows the projected data points achieved by the Sammon’s projection algorithm.

The best performance of the projection performance achieved by the LF algorithm, the DSOM algorithm, and the SIP algorithm from 10 runs are shown in Fig. 8(a)–(c), respectively. These results were achieved by adopting 7×7 artificial agents in each algorithm with proper parameter values. In this example, the proposed SIP algorithm presents 10 line-shaped clusters and the DSOM algorithm grudgingly presents 9 or 10 clusters, but the LF algorithm does not present any clustered groups at all. This example is used to illustrate that the SIP algorithm can produce good results even if the dimensionality of the data set is high (e.g., 20 in this case).

5. Discussion

5.1. The factors affecting the SIP algorithm

In this section, we discuss how the number of doves and the values of parameters affect the SIP algorithm. Due to the limited space, we just present the results for the iris data set. Similar observations can be drawn from the simulations on the remaining data sets. First, we used another three different population sizes to rerun the SIP algorithm to investigate how the swarm size affects the SIP algorithm. Note that the values of the parameters used in the SIP algorithm were set to be the same. Fig. 9 shows the scatter plots achieved by the four different swarm sizes. These four scatter plots look similar and all indicate that the position vectors could be partitioned into one large loose group and one small tight group. This observation implies that the population size does not seriously influence the visualization effect. From many simulations, we have not yet found a strong connection between the number of doves and the number of data points; therefore, some suggestions for choosing the size of doves are given as follows. First, the number of doves should be set to be larger than the possible number of clusters existing in the data set. Second, since a larger size has more the computational load, we suggest the use of a small size of doves (e.g., 7×7 doves).

To investigate how the parameters, λ, η_w, and η_p, affect the SIP algorithm, we used several different combinations with different values of the parameters, λ, η_w, and η_p, to rerun the SIP algorithms for the iris data set. Fig. 10 shows how the value of the parameter, λ, affects the projection performance when the values of the two parameters, η_w and η_p, were fixed. From these simulations, we found that the performance of the SIP algorithm is not sensitive to the value of the parameter, λ. Therefore, we suggest that the value of the parameter, λ, can be set to any value between 0.01 and 0.5.

In the following we investigate how the values of the two parameters, η_w and η_p, affect the projection performance when we fixed
the value of the parameter $\lambda$. The simulation results are shown in Fig. 11. By viewing Fig. 11, we may find that the doves did not move a lot when the values of the parameters, $\eta_w$ and $\eta_p$, were set to be very small (e.g., $\eta_p = \eta_w = 0.01$). The position vectors started to move when the values of the parameters became larger (e.g., $\eta_p = \eta_w > 0.01$). The larger the values the quicker the doves merge together. The doves merged into one group starting from Fig. 11(g); however, if we zoom in the plots shown in Fig. 11(g)–(i) then we will find there are again two separable groups as shown in Fig. 11(j)–(l). The scatter plots achieved by setting the parameters to be from 0.3 to 0.7 look similar. Therefore, we suggest that the values of the parameters, $\eta_w$ and $\eta_p$, can be set to be 0.1.

In the previous section, simulation results demonstrate that our algorithm provides an appealing solution to deal with data sets which consist of well-separated and non-overlapping clusters. From some other simulations we observed that our algorithm sometimes does not work very well for clusters that are not well separated. For example, the data set shown in Fig. 12(a) is consisted of three normally distributed clusters where there is no clear border between clusters. The visualization result achieved by the SIP algorithm with 7×7 doves is shown in Fig. 12(c). Then we adopted the auxiliary method proposed in [54] to trim the data by deleting some data points which lie at the connecting regions among clusters. The auxiliary method first groups the original data set into many clusters and then deletes clusters with low numbers of data points (the number is pre-specified by the user). The trimmed data set is shown in Fig. 12(b). With the same settings of the parameters, the visualization result achieved by the SIP algorithm is shown in Fig. 12(d). Obviously, the visualization result shown in Fig. 12(d) gives us more confidence to partition the position vectors into three clusters than...
the result shown in Fig. 12(c). This example serves to illustrate that the SIP algorithm can provide good results when data sets are consisted of well-separated and non-overlapping clusters.

Here we want to examine whether the initialization method proposed in [52] can improve the performance of the SIP algorithm. Fig. 13 shows the simulation results. The left column, the center column, and the right column are the results at the end of 5 epochs, 10 epochs, and 15 epochs, respectively. Obviously, the SIP algorithm incorporated with the initialization method presents a better visualization effect than the SIP algorithm incorporated with the random initialization method at the end of 5 epochs. The reason can be attributed to the following reasons. By adopting the special initialization method, doves with similar position vectors will be assigned to similar multi-dimensional sense organ vectors. Therefore, neighboring doves will compete for food (i.e., data points) near to them and move in according directions. If neighboring doves are assigned with very different sense organ vectors, $w_j$, then the doves which are near to each other in the two-dimensional artificial ground will have different responses to an input pattern. Understandably, these doves will not move accordingly. Therefore, it is probable that there is no tendency towards the emergence of groups as shown in Fig. 13(b). However, if the number of epochs is large enough (e.g., 15 epochs) both initialization methods present similar visualization effect. One thing should be emphasized is that if a data set is very complex and/or with a large amount of variables then the initialization method proposed in [52] does not guarantee to be able to provide the same improvement effect.

Fig. 14 shows four simulation results to demonstrate whether the performance of the SIP algorithm will be affected by the order of presentation of input patterns. Each simulation was with a different order of presentation. From these simulation results, we find that the patterns of the position vectors look similar and they all present
Fig. 16. The clustering results achieved by the auxiliary method for the SIP algorithm: (a) the data set consisting of two ellipses; (b) the chromosome data set; (c) the iris data set; (d) the breast-cancer data set; (e) the 20-D non-overlapping data set.
two-group structures. Therefore, the performance of the SIP algorithm is not very sensitive to the order of presentation of input patterns for this data set. However, from other simulations, we found that high-dimensional and overlapping data sets, are, to a certain extent, still sensitive to the order of presentation of input patterns. In the implementation of the SIP algorithm, we still keep it a good practice to randomize the order of presentation of input patterns from one epoch to the next.

5.2. An auxiliary method for the SIP algorithm

A complete clustering algorithm involves in two stages. While the first stage is to explore the possible number of clusters existing in the data set, the second stage is to partition the data set into the corresponding number of clusters. The proposed SIP algorithm is a kind of projection algorithm; therefore, similar to other projection algorithms, it relies on human vision to provide the information about the number of clusters.

After we have successfully estimated the number of clusters by visual examination of the scatter plot, we may adopt some other well-developed clustering algorithms (e.g., the k-means algorithm, the FCM algorithm, etc.) to enter the second stage of the clustering phase. In this paper, we propose an auxiliary method to endow the SIP algorithm with the capability of clustering data. To produce the clustering result we subsequently draw a line to dichotomize the region of interest into two sub-regions. At the end of the dichotomizing procedure the M-N doves will be divided into the desired number of groups. We use Fig. 15 to illustrate the dichotomizing procedure. In this example, the number of clusters is estimated to be three. The sense organ vectors of the doves belonging to the same group consist of the templates of the corresponding cluster. For an input pattern, we then find the nearest neighbor among the templates to the input pattern based on the Euclidean distance between the input pattern and the template’s sense organ vector. Finally, the input pattern is assigned to cluster m if the most similar sense organ vector is in group m. For some cases, there are some position vectors which we cannot visually determine which group they should belong to. Under this circumstance, we may temporally not assign them to groups. After all other position vectors have been grouped we then assign these temporally unassigned position vectors to their, respectively, nearest neighboring group based on the Euclidean distance between the position vectors.

The clustering results achieved by the auxiliary method are shown in Fig. 16. One thing should be emphasized is that since the scatter plot achieved by the SIP algorithm indicated that there are only two major clusters (as shown in Fig. 6(c)) in the iris data set, we had no choice but to cluster the iris data set into two clusters. That is, we intentionally merge the two non-separable classes into one large class.

Since these data sets used for examples are consisted of data points with class labels, a solid comparison of the clustering results achieved by different algorithms can be carried out by computing the Jaccard statistic [13,55]. The Jaccard statistic assesses the degree to which two partitions of n objects agree. One partition comes from a clustering result and the second partition is from class labels. The Jaccard statistic is computed as the value of a/(a + b + c) where a is the number of pairs of objects that are in the same group in both partitions, b is the number of pairs of objects that are in the same cluster but with different class labels, and c is the number of pairs of objects that are with the same class label but within different clusters. Thus large values of the Jaccard statistic imply close agreement between the two partitions.

Based on the estimated number of clusters we then ran the k-means algorithm, the FCM algorithm, and the auxiliary method to cluster the data set into the corresponding clusters.

<table>
<thead>
<tr>
<th>Example</th>
<th>Algorithm</th>
<th>Jaccard statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>K-means</td>
<td>8.63e-01 ± 2.58e-02</td>
</tr>
<tr>
<td></td>
<td>FCM</td>
<td>8.26e-01 ± 1.19e-07</td>
</tr>
<tr>
<td></td>
<td>SIP</td>
<td>1.00e+00 ± 0.00e+00</td>
</tr>
<tr>
<td>2</td>
<td>K-means</td>
<td>7.27e-01 ± 9.50e-02</td>
</tr>
<tr>
<td></td>
<td>FCM</td>
<td>7.37e-01 ± 1.98e-01</td>
</tr>
<tr>
<td></td>
<td>SIP</td>
<td>1.00e+00 ± 0.00e+00</td>
</tr>
<tr>
<td>3</td>
<td>K-means</td>
<td>9.30e-01 ± 1.19e-07</td>
</tr>
<tr>
<td></td>
<td>FCM</td>
<td>9.30e-01 ± 1.19e-07</td>
</tr>
<tr>
<td></td>
<td>SIP</td>
<td>1.00e+00 ± 0.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>K-means</td>
<td>8.65e-01 ± 1.78e-07</td>
</tr>
<tr>
<td></td>
<td>FCM</td>
<td>8.49e-01 ± 1.78e-07</td>
</tr>
<tr>
<td></td>
<td>SIP</td>
<td>8.64e-01 ± 1.97e-02</td>
</tr>
<tr>
<td>5</td>
<td>K-means</td>
<td>7.90e-01 ± 1.00e-01</td>
</tr>
<tr>
<td></td>
<td>FCM</td>
<td>9.45e-01 ± 1.67e-01</td>
</tr>
<tr>
<td></td>
<td>SIP</td>
<td>1.00e+00 ± 0.00e+00</td>
</tr>
</tbody>
</table>

Each algorithm was conducted for 10 runs using different presentation orders and/or combinations of parameters. Table 1 tabulates the performance achieved by these three algorithms. The mean value and the standard deviation value represent the mean and the standard deviation of the performance of the 10 runs. The highlighted (bold) entries correspond to best solutions found by these algorithms. From Table 1, we observe that the SIP algorithm incorporated with the auxiliary method was able to achieve 100% correct results if the clusters are well separated. As for the Wisconsin breast cancer data set, the FCM outperformed the SIP algorithm incorporated with the auxiliary method since Fig. 8(c) did not present two well-separated clusters. Furthermore, if the user did not dichotomized the position vectors in the way as shown in Fig. 16(d) then the Jaccard statistic achieved by the SIP algorithm will be even lower.

6. Conclusion

In this paper, a new kind of swarm-inspired projection algorithm is proposed. The development of this new algorithm was motivated by the foraging behavior of doves. The focus of the paper is not to precisely model the doves’ foraging behavior, but to show that some basic foraging principles may serve as metaphors for data clustering. The SIP algorithm can project high-dimensional data points into a 2-dimensional data space. The scatter plots of position vectors p_j may facilitate visual inspection of high-dimensional data.

From many simulations, we find the performance is not very sensitive to the sizes of the doves as is demonstrated by testing the algorithm on the iris data set. For the time being, our algorithm still remains a lot of space to be improved to become more robust and fault tolerant.
an effective data analysis tool, in this paper, we present an interesting approach to utilizing some basic doves’ foraging principles as metaphors for data clustering.

Acknowledgement

This work was partly supported by the National Science Council, Taiwan, ROC, under NSC 97-2631-S-008-003 and NSC 97-2631-H-008-001.

References

About the Author—SHI-YONG SU received the M.S. degree in computer science and information engineering at National Central University, Taiwan, in 2007. Her research interests include neural networks, pattern recognition, and human–computer interfaces.

About the Author—Yu-Xiang Zhao received the B.S. and M.S. degrees in electrical engineering from Tamkang University, Taiwan, in 2000 and 2002, respectively, and the Ph.D. degree in computer science and information engineering from National Central University, Taiwan, in 2007. He is currently an assistant professor of computer science and information engineering at Ta Hwa Institute of Technology, Taiwan. His research interests include neural networks, pattern recognition, swarm intelligence, and image processing.