IMPLEMENTATION OF WEIGHTED KERNEL DETERMINISTIC ANNEALING ALGORITHM FOR NON-LINEAR SHAPE CLUSTERING

A THESIS

Submitted by

SUSHANT DOGRA
EE13B107

In the partial fulfilment of the requirements
for the award of the degrees of

BACHELOR OF TECHNOLOGY
and
MASTER OF TECHNOLOGY

Department of Electrical Engineering
Indian Institute of Technology Madras

May 2018

THESIS CERTIFICATE

This is to certify that the thesis titled "**Implementation of Weighted Kernel Deterministic Annealing algorithm for non-linear shape clustering**", submitted by Sushant Dogra (EE13B107), to Indian Institute of Technology Madras, in partial fulfilment of the requirements for the award of the degrees of **Bachelor of Technology** and **Master of Technology** in **Electrical Engineering**, is a bona fide record of the work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Bharath Bhikkaji

Project Advisor

Assistant Professor

Dept. of Electrical Engineering

IIT Madras, 600 036

Place: Chennai

Date: 10 May 2018

ACKNOWLEDGEMENT

I express my sincere gratitude to **Prof. Bharath Bhikkaji** for his indispensable guidance and constant encouragement which made the endeavour to complete my project fruitful experience. His diligent and industrious disposition always inspired to persevere with my work and complete it successfully.

I would like to thank Dr. Arun Ayyar and Shravan who constantly supported me, provided me with insightful inputs and advised me when I found myself stuck. Without their contribution this project would not have been realized.

I would also like to thank my family and friends who have always supported me in all my endeavours with their heart and soul. It is their constant love and motivation that made it possible for me to stay here and complete this project.

ABSTRACT

KEYWORDS: Clustering Analysis, K-means algorithm, Deterministic Annealing, Maximum Entropy Principle, Shannon Entropy, Kernels

Clustering analysis is extensively used to partition data into different sets such that data points in the same set are more similar to each other than data points from different sets. From a mathematical point of view, clustering is a non-convex optimisation problem. K-means clustering is one of the simplest algorithms used for partitioning data that is linearly separable in the input space. But it is sensitive to initialisation, and often returns sub-optimal solutions. The deterministic annealing (DA) approach to clustering demonstrates substantial improvement over K-means method. It is derived within a probabilistic framework from basic information theoretic principles (e.g. maximum entropy principle). The application specific cost is minimised subject to a constraint on the randomness (Shannon entropy) of the solution, which is gradually lowered. We emphasize on intuition gained from analogy to physical chemistry, where the annealing process avoids many shallow local minima of the specified cost and, at the limit of zero "temperature", produces a non-random (hard) solution. We further look into a modification of deterministic annealing approach called weighted kernel deterministic annealing (WKDA). This algorithm combines the kernel trick with distributed aspect of the deterministic annealing algorithm to produce effective clustering solutions that are independent of initialisation and has the ability to partition data that is not linearly separable in the input space in the desired way. Finally, we test the WKDA algorithm on several standard test cases to check its performance.

TABLE OF CONTENT

LIST OF FIGURES

NOTATIONS

N	Number of input points
M	Dimension of each input point
K	Number of clusters
$\mathcal{X}$	Set of all input points
$\mathcal{Y}$	Set of all cluster centroids
$p(x_i)$	Relative significance of point x_i
$d(x, y)$	Squared Euclidian distance between points x and y
$p(y\|x)$	Association probability of input vector x to the code vector y
$H(\mathcal{X}, \mathcal{Y})$	Shannon entropy of the system
T	Initial Temperature of the system
T_{min}	Minimum Temperature allowed in the system
α	Cooling rate of the system
$< x, y >$	Inner product of vectors x and y
σ	Standard deviation of the Gaussian kernel

CHAPTER 1

INTRODUCTION

Cluster analysis or clustering is a type of unsupervised learning method and has emerged as one of the fundamental problem in data mining in recent years. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labelled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples. The complementary category of supervised learning involves a "teacher" who provides, during the training phase, the desired output for each input sample.

Clustering is a task of dividing the population or data points into a number of groups/partitions such that data points in the same partition are more similar to each other and dissimilar to the data points in other partitions. The similarity measures are based upon metrics such as Euclidean, Manhattan and Bergman divergences. They represent distances of data points from their corresponding cluster centroids, or pairwise distances between any two data points in the input space.

For example: We can distinguish the data points given in the figure below into three different clusters and can say that the points which belong to the same cluster are more similar to each other than points that belong in different clusters.

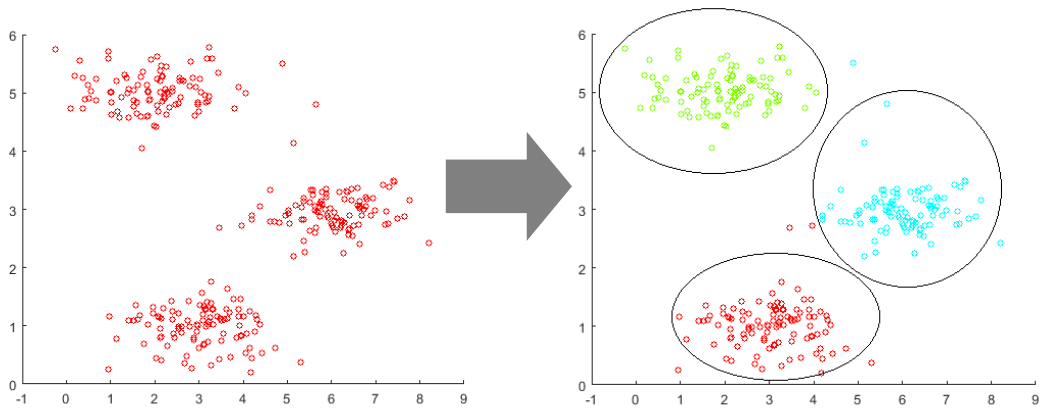


Figure 1 – Basic example of clustering

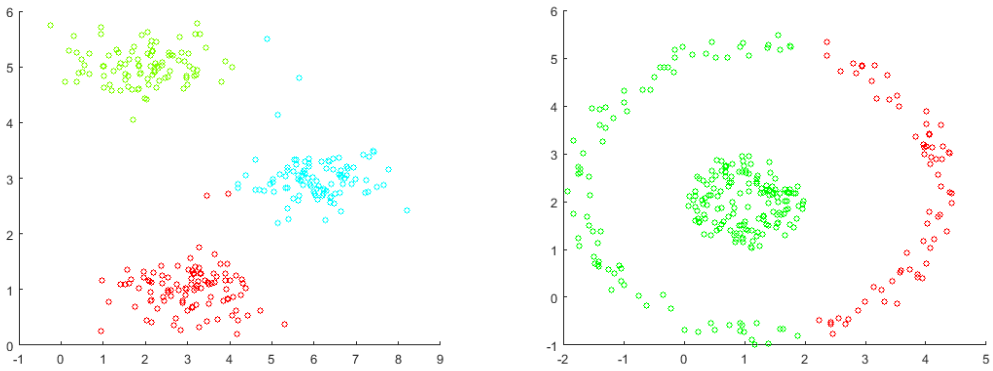
The task of clustering is computationally difficult and the design of a practical system must take into account its complexity. Thus, typically, we restrict the complexity of the

clustering algorithms by specifying K, i.e. the number of partitions that we want to create in the input data a priori. Mathematically, clustering is a non-convex optimisation problem.

A particularly well known and simplest approximation method for clustering is known as the "k-means algorithm". It was first proposed by *Macqueen* [1] in 1967. It is an unsupervised, non-deterministic, numerical, iterative method of clustering. The main idea of this algorithm is to, initially, randomly define K centroids, one for each cluster, and associate all the data points to their nearest centroid. Then, iteratively improve the location of the centroids so as to reach a local optimum. One of the main drawbacks of this algorithm is that it is sensitive to the selection of initial centroids and often gets stuck at local optimums. Thus, we run this algorithm multiple times and return the best solution.

To overcome this curse of initialisation, *Rose* [4] proposed an annealing based algorithm in 1998, well described in terms of laws, such as, maximum entropy principle (MEP) in statistical physics literature, and showed that the solutions obtained using this approach are totally independent of the choice of initial configurations. This algorithm is referred as "Deterministic Annealing (DA) algorithm" and is aimed to provide high quality solutions to clustering problems with only marginal increase in computational complexity. The observation of annealing process in physical chemistry motivated the use of similar concepts to avoid local minima of the cost function.

A major drawback of both K-means and DA algorithms is their incapability to separate clusters that are not linearly separable in the input space. Figure 2 shows the performance of DA algorithm in identifying the natural clusters for two distinct sets of data points.



(a) Linearly Separable Data *(b) Non-linearly Separable Data*

Figure 2 – Example of clustering by DA algorithm on (a) linearly separable data, and (b) non-linearly separable data

While the data points in Figure 2(a) can be separated using hyper planes in $\mathbb{R}^2$ (line), there is no such line that can separate data points distributed along two concentric circles in Figure 2(b). Thus, while the DA algorithm finds the optimal linear separation of data points in Figure 2(b), such separations are indeed not natural and often undesired.

To overcome these limitations of K-means and DA algorithms, a novel "Weighted Kernel Deterministic Annealing" approach is presented by *Mayank Banarwal* [8] in his paper. The WKDA algorithm is based on the Deterministic Annealing algorithm presented by *Rose* but also uses the kernel-trick to map data points to higher dimensional space and then clusters data points using linear separators in the new space. The WKDA algorithm enjoys the best of both worlds. On one hand, the algorithm is independent of initialisation like the basic DA algorithm; and on the other hand, WKDA does not require a lot of computational power. Furthermore, by modifying the algorithm suitably, we can easily implement must-link and cannot-link constraints between any two points.

(a) Clustering by DA algorithm *(b) Clustering by WKDA algorithm*

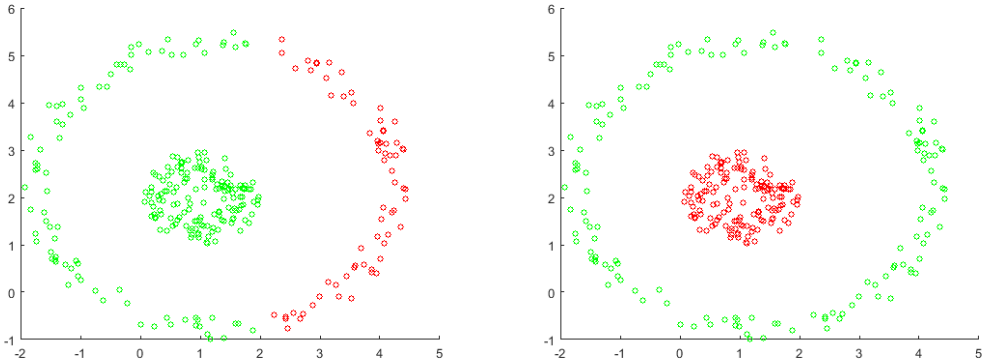


Figure 3 – Example of clustering by (a) DA algorithm, and (b) WKDA algorithm on non-linearly separable data

As it as be seen from Figure 3, the WKDA algorithm divides the data points in a more desired way (as in Fig. 3b) as compared to the basic DA algorithm (as in Fig. 3a).

This thesis is organised as follows. Chapter 2 explains the mathematical formulation of the clustering problem and goes on to explain the k-means algorithm and the results obtained from it. Chapter 3 introduces the basic deterministic annealing algorithm, wherein we first introduce the mathematical formulation of the clustering problem in a slightly different way from chapter 2 and go on to solve the formulated problem to reach to a basic algorithm. Chapter 4 further modifies the previously introduced DA algorithm by using the kernel trick for shape clustering applications and introduces the WKDA algorithm. Chapter 5 wraps up this thesis by evaluating the WKDA algorithm on few example scenarios and giving conclusions and results.

CHAPTER 2

K-MEANS CLUSTERING

2.1 INTRODUCTION

K-means is one of the simplest unsupervised learning algorithm that solves the clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assuming K clusters) fixed a priori. The main idea is to randomly initialise K centroids, one for each cluster. After initialisation, the next step is to take each point belonging to the given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and early grouping is done. At this point, we need to re-calculate K new centroids as barycentre of the clusters resulting from the previous step. After we have these K new centroids, a new binding has to be done between the same data set points and the nearest new centre. A loop has been generated. As a result of this loop we may notice that the k centres change their location step by step until no more changes are done or in other words centres do not move any more.

2.2 MATHEMATICAL FRAMEWORK

Given a set of $N \in \mathbb{N}$ points $\mathcal{X} = \{ x_i : x_i \in \mathbb{R}^M , 1 \leq i \leq N \}$, the objective is to find the optimal locations of $K \in \mathbb{N}$ cluster centroids denoted by $\mathcal{Y} = \{ y_j : y_i \in \mathbb{R}^M , 1 \leq j \leq K \}$ such that the aggregated sum of distances of each point from its nearest cluster centroid is minimized. The objective function (also known as the squared error function or the cost function) that this algorithm aims to minimize is given as follows:

$$J = \min_{\{y\},\{t\}} \sum_{j=1}^{K} \sum_{i=1}^{N} t_{ij} . p(x_i) . d(x_i, y_j)$$

Here, $t_{ij} = \begin{cases} 1 & if\ point\ i\ is\ assigned\ to\ cluster\ j \\ 0 & otherwise \end{cases}$

$p(x_i)\ is\ the\ relative\ significance\ of\ point\ x_i\ (\approx \frac{1}{N}\ , generally)$

$d(x_i, y_j) = \sum_{q=1}^{M} \left(x_i^q - y_j^q \right)^2\ ,\ i.e.\ it\ is\ the\ sqaured\ euclidean\ distance$

2.3 PSEUDO CODE FOR THE ALGORITHM

Step 1: Randomly select K cluster centroids.

Step 2: Calculate the distance between each data point and all cluster centroids using:

$$d(x_i, y_j) = \sum_{q=1}^{M} \left(x_i^q - y_j^q\right)^2 \quad \forall \; i \in \{1, \dots, \mathrm{N}\}, \mathrm{j} \in \{1, \dots, \mathrm{K}\}$$

Step 3: Assign the data points to the cluster centre whose distance from the cluster centre is minimum of all cluster centres.

Step 4: Recalculate the new cluster centres using:

$$y_j = \frac{1}{C_j} \sum_{q=1}^{C_j} x_q \quad \forall \; \mathrm{j} \in \{1, \dots, \mathrm{K}\}$$

$$here, C_j \; represents \; the \; number \; of \; data \; points \; in \; cluster \; j \; and$$

$$x_1, \dots\dots, x_{C_j} \; are \; the \; points \; assigned \; to \; cluster \; j \; in \; the \; previous \; step.$$

Step 5: Recalculate the distance between each data point and new obtained cluster centres.

Step 6: If no values change then stop, otherwise repeat from step 3.

2.4 ANALYSIS OF THE ALGORITHM

The K-means algorithm is fairly easy and intuitive to understand. It is quite efficient with the time complexity of the order $O(q.k.n.m)$, where n is the number of data points, k is the number of clusters, m is the dimension of each data point and q is the number of iterations. It can be easily shown that the K-means algorithm converges by proving that its cost function monotonically decreases. The figure below demonstrates the same by plotting value of the cost function against the number of iterations, and it can be observed that the cost monotonically decreases until we reach a local minimum.

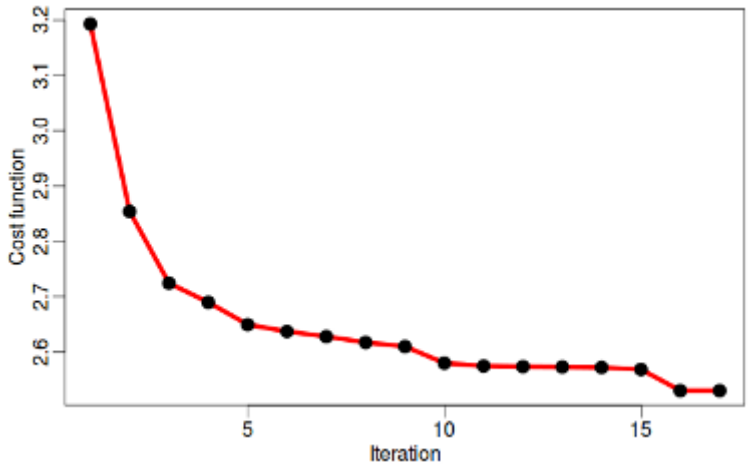


Figure 4 – Plot of value of the cost function against number of iterations

Despite having many advantages, there are several limitations of k-means algorithm which makes it unsuitable for complex cases. One of main limitation of this algorithm is that it fails to obtain desired partitions of the input space when the clusters are not linearly separable.

Also, it can be easily seen that the objective function for k-means is a non-convex function. Thus, the cost surface of the objective function is riddled with several local optimums and since, there is no prescribed way of initialising the algorithm, upon random initialisation, it frequently happens that suboptimal partitions are found. Thus, we need to run the algorithm multiple times to reach the solution with optimal partitions. An example of the above described limitation is given below:

a) Result after the first run *(b) Result after the second run*

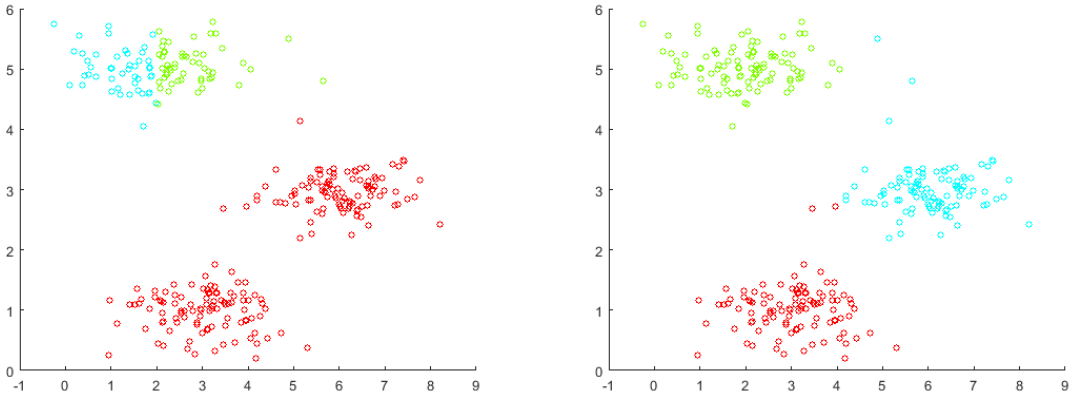


Figure 5 – Results of K-means clustering after the (a) first run, and (b) second run of the algorithm

As it can be seen from the figure 5(a), after the first run of the code, the k-means algorithm gets stuck at a local minimum which has high error and the clusters obtained are sub-optimal .We reach the global minimum after the second run of the code as can be seen in figure 5(b). The mean squared error of the partitions in figure 5(a) is also higher than the mean squared error of the partitions in figure 5(b).

To overcome this curse of random initialisation, *Rose* [4] proposed an annealing based algorithm called Deterministic Annealing which is discussed in the next chapter.

CHAPTER 3

DETERMINISTIC ANNEALING

3.1 INTRODUCTION

The deterministic annealing (DA) method is used for solving several non-convex optimisation problems because of its ability to avoid shallow local minimums of a given cost function even when there are several local optimums. It has demonstrated substantial performance improvements over standard learning methods in a variety of important applications including compression, clustering estimation, classification and statistical regression. It is established in a probabilistic framework through basic information-theoretic techniques such as Maximum Entropy Principle (MEP) and random coding.

The observation of the annealing processes in physical chemistry motivated the use of similar concepts to avoid local minimums of the optimisation cost. Certain chemical systems can be driven to their low energy state by annealing, which is a process wherein we start at very high temperatures and gradually reduce it, spending a lot of time at the vicinity of the phase transition points to reach the desired state. In the corresponding probabilistic framework, a distribution is defined over the set of all the possible configurations which assigns higher probability to configurations of lower energy. This distribution is parameterized by "temperature", and as the temperature is lowered, it becomes more and more discriminating (concentrating most of the probability in a smaller subset of low energy configurations). At the limit of low temperature, it assigns non-zero probability only to global minimum configurations.

Basic clustering methods suffer from the problem of poor local minima that riddle the cost surface. A variety of heuristic approaches have been proposed to tackle this difficulty, and they range from repeated optimisation with different initialisation, and heuristics to obtain good initialisation, to heuristic rules for cluster splits and merges etc. But DA looks at the clustering problem from a different viewpoint and takes a very different approach for solving the problem. The approach is based on information theory and probability, and consists of minimising the clustering cost at prescribed levels of randomness (Shannon entropy). It provides clustering solutions at different scales, where the scale is directly related to the temperature parameter.

3.2 MATHEMATICAL FRAMEWORK

We use the following terminology that applies to both signal processing and clustering setups. We assume that we have a source that produces a sequence of independent and identically distributed input vectors $x_1, x_2, x_3, \ldots\ldots, x_N$ ($x_i \in \mathbb{R}^M, 1 \leq i \leq N$) according to some probability distribution $p(x)$. We also assume that there exists an encoding function $y(x)$ that maps the input vector $\mathcal{X}$ to the best reproduction code vector in some finite set $\mathcal{Y}$. In the clustering setup, the input vectors correspond to the training set, while $\mathcal{Y}$ corresponds to the set of some appropriately defined cluster centroids. The distance between an input vector and code vector assigned to it is called as distortion. The aim is to minimize the average distortion for a given set of input vectors. Thus, the objective function is given by:

$$D = \sum_{i=1}^{N} p(x_i).d(x_i, y(x_i))$$

$here, distortion\ \ d(x,y) = \sum_{q=1}^{M}(x^q - y^q)^2\ , i.e.\ it\ is\ the\ sqaured\ euclidean\ distance.$

Most algorithms for clustering (such as the K-means algorithm) start with some initial values of the code vector $\mathcal{Y}$ and iteratively optimize over them as the algorithm proceeds. However, such approaches as sensitive to the choice of initial values and primarily die due to the distributed aspect of the clustering problem, where any change on the coordinates of x_i affects the distortion $d(x_i, y(x_i))$ only with respect to the nearest code vector. The DA algorithm overcomes this sensitivity by allowing fuzzy association of every data point to each code vector.

A probabilistic frame work for clustering is defined here by randomisation of the partition, or equivalently randomisation of the encoding rule. The input vectors (or the input data points) are assigned to clusters in probability, which is called as the association probability. This viewpoint bears similarity to "fuzzy" clustering, where each data point has partial membership of clusters. This is different from K-means clustering, which is similar to "hard" clustering, where each data point is associated to only one cluster at a time.

However, this formulation is purely probabilistic. While it considers clusters as regular (non-fuzzy) sets whose exact membership is the outcome of a random experiment, one may also consider the fuzzy sets obtained by equating the degree of membership with the association probability in the probabilistic model. Thus, the traditional frame work for clustering is the marginal special case where all association probabilities are either zero or one.

For randomisation part, the modified average distortion function can be written as:

$$D = \sum_{x}\sum_{y} p(x,y).d(x,y)$$

$here, distortion\ \ d(x,y) = \sum_{q=1}^{M}(x^q - y^q)^2\ \ , i.e.\ it\ is\ the\ sqaured\ euclidean\ distance.$

$and\ \ p(x,y)\ is\ the\ joint\ probability\ distribution\ defined\ on\ \mathcal{X} * \mathcal{Y}.$

We can rewrite the above expression as:

$$D = \sum_{x} p(x) \sum_{y} p(y|x).d(x,y)$$

$here, p(y|x)\ is\ the\ association\ probability\ relating\ input\ vector\ x\ with\ code\ vector\ y$

The probability distribution $\{p(y|x)\}$ assesses the trade-off between decreasing local influence and the deviation of the modified average distortion function from the original average distortion function.

At the limit where the association probabilities are "hard" (either 0 or 1) and each input vector is assigned to a unique code vector with probability one, the DA distortion expression given above becomes identical to the hard clustering distortion function.

Minimisation of D with respect to the free parameters $\{y, p(y|x)\}$ would immediately produce a hard clustering solution, as it is always advantageous to fully assign an input vector with probability one, to the nearest code vector. However, we recast this optimisation problem as that of seeking the probability distribution which minimises D subject to a specified level of randomness. The level of randomness is, naturally, measured by Shannon entropy. It captures the uncertainty in associating input data points with the code vectors. The expression of entropy for our system is given as:

$$H(\mathcal{X},\mathcal{Y}) = -\sum_{x}\sum_{y} p(x,y).log(p(x,y))$$

Now, the optimisation is conveniently reformulated as minimisation of the Lagrangian:

$$F = D - T.H$$

Here, T is the Lagrange multiplier, D is the average distortion and H is the Shannon entropy. To further analyse this Lagrangian F in the above stated equation, we note that the joint entropy of the system can be further decomposed into two terms:

$$H(\mathcal{X},\mathcal{Y}) = H(\mathcal{X}) + H(\mathcal{Y}|\mathcal{X})$$

$$H(\mathcal{X},\mathcal{Y}) = -\sum_{x} p(x).\log\big(p(x)\big) - \sum_{x} p(x) \sum_{y} p(y|x).\log(p(y|x))$$

Here, note that the first term is the source entropy, which is independent of clustering, thus, we can ignore that term while optimisation and focus on conditional entropy. Thus, the Lagrangian that we need to minimise becomes

$$F = \sum_{x} p(x) \sum_{y} p(y|x).d(x,y) \; - T.(-\sum_{x} p(x) \sum_{y} p(y|x).\log\big(p(y|x)\big))$$

Clearly, for large values of T we mainly attempt to maximize the entropy. As T is lowered, we trade entropy for reduction in distortion, and as T approaches zero, we minimise D directly to obtain a hard (non-random) solution.

At this point, let us consider an equivalent derivation of the above obtained Lagrangian based on the principle of maximum entropy. Suppose we fix the level of expected distortion D, and seek to estimate the underlying probability distribution. The objective is to characterise the random solution at gradually diminishing levels of distortion until minimum distortion is reached. The maximum entropy principle states that "Among all the probability distributions that satisfy a given set of constraints, choose the one that maximizes the entropy (uncertainty)". The informal justification is that while this choice agrees with what is known (the given constraints), it maintains maximum uncertainty with respect to everything else. Had we chosen another distribution satisfying the constraints, we would have reduced the uncertainty and would have therefore implicitly made some extra restrictive assumption.

For the problem described above, we seek a distribution that maximises the Shannon entropy while satisfying the expected distortion constraint. The corresponding Lagrangian to maximize is $H - \beta.D$, with β as the Lagrangian multiplier. The equivalence of the two derivation is obvious, and both Lagrangians are simultaneously optimised by the same solution configuration for $\beta = 1/T$.

Maximising the entropy is commensurate with decreasing the local influence. The trade-off between maximising the entropy and minimising the modified distortion function is addressed by seeking the probability distribution $\{p(y_j|x_i)\}$ that minimises the Lagrangian. Solving the Lagrangian and thus minimising F with respect to the association probabilities $p(y|x)$ gives us the Gibbs distribution (which is the partition function of statistical physics):

$$p(y_j|x_i) = \frac{\exp(^{-d(x_i,y_j)}/_{T})}{Z_{x_i}}$$

Where, the normalisation is

$$Z_{x_i} = \sum_{j=1}^{K} \exp(^{-d(x_i,y_j)}/_{T})$$

The corresponding minimum of F is obtained by plugging in the expression of association probabilities back in the Lagrangian.

$$\begin{aligned} F^* &= \min_{\{p(y|x)\}} F \\ &= -T.\sum_x p(x).\log(Z_x) \\ &= -T.\sum_x p(x).\log(\sum_y \exp\left(^{-d(x,y)}/_{T}\right)) \end{aligned}$$

To find the optimal locations of the centroids, we minimise the Lagrangian with respect to the code vector locations $\{y\}$, its gradients are set to zero yielding the condition

$$\sum_x p(x,y)\frac{d}{dx}d(x,y) = 0 \qquad \forall y \in \mathcal{Y}$$

After normalisation by $p(y) = \sum_x p(x,y)$, the condition can be written as a centroid condition

$$\sum_x p(x|y)\frac{d}{dx}d(x,y) = 0 \qquad \forall y \in \mathcal{Y}$$

Here, $p(x|y)$ denotes the posterior probability calculated by using the Bayes' rule, which for the squared error distortion function gives the results

$$y_j = \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).x_i}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)}$$

The above equation has a form similar to computing centroids in the K-means clustering algorithm. However in K-means clustering, the associations between x_i and y_j as hard (0-1). The DA algorithm alternates between the two steps of calculating the association probabilities $\{p(y_j|x_i)\}$ and calculating the code vectors $\{y_j\}$ at each T until convergence. In fact, the convergence of the algorithm is guaranteed as a consequence of coordinate decent on the Lagrangian.

3.3 PSEUDO CODE FOR THE ALGORITHM

Step 1: Initialise all the input parameters of the algorithm, namely – Number of clusters (K), Initial Temperature (T), Minimum Temperature (T_{min}), Cooling rate (α), Convergence margin ($error$).

Step 2: Let N be the number of data points, M be the dimension of each data point, and K be the number of clusters. Initialise the – association probability matrix (size: NxK) with zeros, and randomly assign K input data points to code vectors y. Unless specifically stated, assign $p(x_i) = {}^{1}/_{N} \; \forall$ i ∈ {1, … , N}.

Step 3: Run the loop given below:

While ($T > T_{min}$) {

While (true) {

Step 3(a): Calculate all association probabilities using

$$p(y_j|x_i) = \frac{\exp({}^{-d(x_i,y_j)}/_{T})}{Z_{x_i}}$$

Where, $Z_{x_i} = \sum_{j=1}^{K} \exp({}^{-d(x_i,y_j)}/_{T}) \; \forall$ i ∈ {1, . . , N} , j ∈ {1, . . , K}

And $d(x_i, y_j) = \sum_{q=1}^{M}(x_i^q - y_j^q)^2$

Step 3(b): Calculate the new values of code vectors using

$$y_j = \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).x_i}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)}$$

$$\forall \; j \in [1, \dots , K]$$

If (Converged) break;

}

$T = T * \alpha$; /* the cooling process */

}

Step 4: Assign clusters to data points using the association probability matrix.

3.4 ANALYSIS OF THE ALGORITHM

The DA algorithm consists of minimising F^* with respect to the code vectors, starting at high value of T and tracking the minimum while lowering T. The central iteration consists of two steps:

(a) Fixing the code vectors and computing the association probabilities.

(b) Fixing the association probabilities and optimising the code vectors.

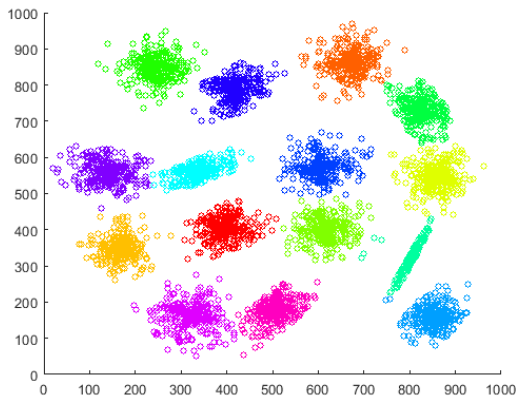
Clearly, the procedure is monotone non-increasing in F^* and converges to a minimum. At high levels of T, the cost is very smooth and, under mild assumptions, can be shown to be convex, which implies that the global minimum of F^* is found. As T tends to zero the association probabilities become binary and a hard clustering solution is obtained.

Some intuitive notion of the workings of the system can be obtained from observing the evolution of the association probabilities. At infinite T, these are uniform distributions, i.e., each input vector is equally associated with all code vectors. These are extremely fuzzy associations. As T is lowered, the distributions become more discriminating and the associations less fuzzy. At the temperature tends to zero, the classification is hard with each input sample assigned to the nearest code vector with probability one. This is the condition in which traditional techniques such as K-means algorithm work. From the DA viewpoint, standard methods are "zero temperature" methods. It is easy to visualize how the zero temperature system cannot "sense" a better optimum farther away, as each data point exercises its influence only on the nearest code vector. On the other hand, by starting at a higher temperature and slowly "cooling" the system, we start with each data point equally influencing all code vectors and gradually localize the influence. This gives us some intuition as to how the system senses, and settles into, a better optimum.

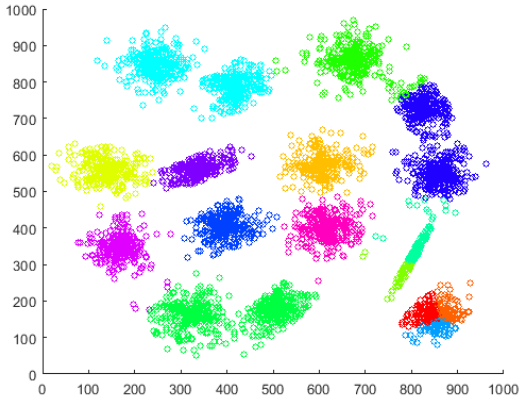
There are several input parameters for this algorithm such as $T_{initial}, T_{min}$ etc. that one must carefully select according to the input data to run the algorithm successfully. For clustering analysis, we have focused on the geometric cooling law $T(t) = \alpha.T(t+1), 0 < \alpha < 1$, where $T(t)$ is the temperature at iteration t and we must also be very careful with the annealing schedule, i.e., the rate at which the temperature is lowered.

The DA algorithm has the worst case time complexity of $O(n.k^2.m.p.q)$, where n is the number of data points, k is the number of clusters, m is the dimension of each data point, p is the number of iterations in the outer loop, and q is the maximum number of iterations in the inner loop.

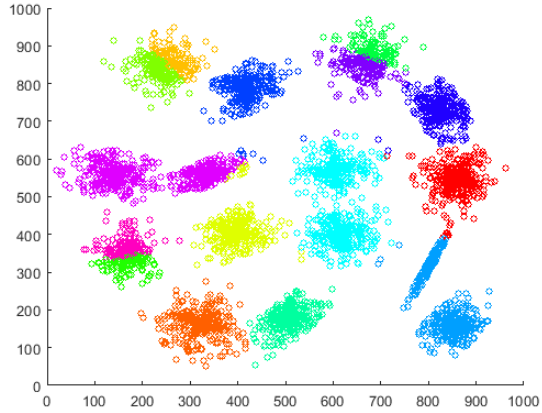
Although the DA algorithm has a higher time complexity as compared to the K-means algorithm, it proves out to be advantageous over simpler algorithms in cases where the complexity (number of data points, dimension, or number of clusters) of the input data set is higher. DA algorithm reaches the optimum in one run whereas algorithm like K-means tend to take several runs to hit the global minimum. For a 2-dimensional data set with $N = 500$ data points and K = 15 clusters, the results of both the algorithms can be seen below:



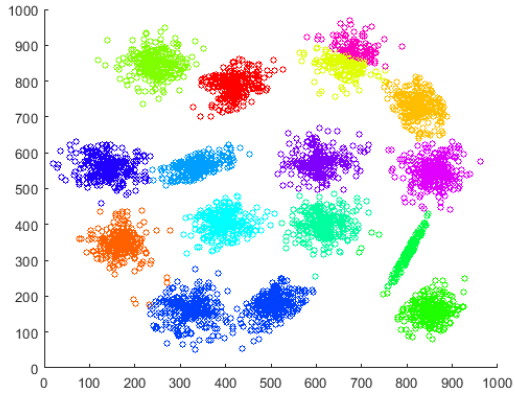
(a) Clustering by DA algorithm



(b) Clustering by K-means algorithm (1st Run)



(c) Clustering by K-means algorithm (2nd Run)



(d) Clustering by K-means algorithm (3rd Run)

Figure 6 – Results of DA and K-means clustering

As it can be seen from the figure 6(a), whereas the DA algorithm hits the global minimum of the cost surface just after its first run and makes the desired partitions of the input space, the K-means algorithm tends to get stuck at the local minimums of the cost surface as in figure 6(b), 6(c) and 6(d), thus giving sub-optimal clustering results.

Although the DA algorithm has overcome the curse of initialisation, it is still limited by linear decision boundaries and can't partition the data that is not linearly separable in the desired way. To overcome this limitation, *Mayank Banarwal* [8] proposed a modification in the DA algorithm which is presented in the next chapter.

CHAPTER 4

WEIGHTED KERNEL DETERMINISTIC ANNEALING

4.1 INTRODUCTION

The Weighted Kernel Deterministic Annealing (WKDA) algorithm is an extension of the basic DA algorithm for shape clustering scenarios as shown in Figure 3(b).It has slightly higher worst case time complexity than the basic DA algorithm but is capable of partitioning data that is not linearly separable in the input space. This is achieved by mapping the data $\mathcal{X}$ in the input space to a higher dimensional feature space where the data becomes linearly separable through an appropriate choice of kernel functions. The mapping allows the use of linear separators to extract clusters in the implicit feature space. Thus, using kernels along with the basic DA algorithm enables it to operate in a higher dimensional feature space without ever explicitly computing the coordinates of the input data points in that space. This approach is referred as "kernel trick" and is often used in several machine learning applications such as support vector machine (SVM), kernel perceptron, clustering, ridge regression, principle component analysis (PCA) etc.

4.2 MATHEMATICAL FRAMEWORK

Before we start working on the mathematics for the WKDA algorithm, let us look into two new concepts, namely – inner products and kernels, which will later help us in understanding the WKDA algorithm better.

Inner Product: It is an algebraic operation that takes two equal length sequences of numbers and returns a scalar number which is the sum of the products of the corresponding entries of the two sequences of numbers. For example, let x, y be two vectors of size M each. Then the inner product of these two vectors is given by:

$$< x, y > = \sum_{i=1}^{M} x^i . y^i$$

Kernels: The kernel functions, also called the similarity functions are used to calculate how similar to two input vectors are. They are used to calculate the inner product of vectors in a higher dimensional input space without explicitly calculating the vectors in the higher dimensional space.

Let x_i , $x_{i'}$ be two input vectors. Rather than first going from the input space ($\mathcal{X}$) to a higher dimensional space (H) using some non-linear feature map $\phi : \mathcal{X} \to H$, and then calculating the inner product in the higher dimensional space, we directly compute the inner product using a kernel function $k(x_i, x_{i'})$, such that

$$k(x_i, x_{i'}) = < \phi(x_i), \phi(x_{i'}) >_H$$

For a given set of data points $\mathcal{X} = \{x_i : x_i \in \mathbb{R}^M , 1 \leq i \leq N\}$, a kernel matrix $K \in \mathbb{R}^{NxN}$ is given by $K_{ii'} = k(x_i, x_{i'}) \; \forall$ i, i′ $\in \{1, .., \mathrm{N}\}$. While the explicit representation of ϕ is not necessary, its existence is guaranteed as long as k satisfies the Mercer's condition (The condition requires that K must be a positive semi-definite matrix).

Many popular choices of kernel functions exist, and the kernel function that we have used here for the implementation of the WKDA algorithm is called the Gaussian kernel. The formula for calculating the Gaussian kernel is given below

$$k(x_i, x_{i'}) = \exp\left(\frac{-d(x_i, x_{i'})}{2.\sigma^2}\right) where, \qquad d(x_i, x_{i'}) = \sum_{q=1}^{M} \left(x_i^q - x_{i'}^q\right)^2$$

Here, σ is called as the standard deviation and it determines the width of the kernel.

Now that we have the understanding of all the necessary concepts, we look into the modifications done to the basic DA algorithm that enable it to partition non-linearly separable data in the desired way.

Note that one of the key steps while implementing the DA algorithm is calculating the distortion between all the data points and cluster centroids. This is written as:

$$d\left(x_i, y_j\right) = \sum_{q=1}^{M} \left(x_i^q - y_j^q\right)^2$$

We can rewrite the above given expression using the inner-products as:

$$d\left(x_i, y_j\right) = < x_i, x_i > \; - 2 < x_i, y_j > \; + < y_j, y_j >$$

Now, if we map our input data to a higher dimensional feature space using a non-linear feature map ϕ, and calculate the distortion between the data point $\phi(x_i)$ and cluster centroid y_j in the implicit feature space, the expression for distortion is given as:

$$d\left(\phi(x_i), y_j\right) = < \phi(x_i), \phi(x_i) > \; - 2 < \phi(x_i), y_j > \; + < y_j, y_j >$$

$$where, \qquad y_j = \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).\phi(x_i)}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)}$$

Each of the three terms in the distortion function given above can be rewritten such that we don't have to explicitly calculate the value of y_j.

- $< \phi(x_i), \phi(x_i) > = K_{ii}$
- $< \phi(x_i), y_j > = < \phi(x_i), \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).\phi(x_i)}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)} >$

$$= \frac{\sum_l p(x_l).p(y_j|x_l).K_{il}}{\sum_l p(x_l).p(y_j|x_l)}$$

- $< y_j, y_j > = < \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).\phi(x_i)}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)}, \frac{\sum_{i=1}^{N} p(x_i).p(y_j|x_i).\phi(x_i)}{\sum_{i=1}^{N} p(x_i).p(y_j|x_i)} >$

$$= \frac{\sum_{l,m} p(x_l).p(x_m).p(y_j|x_l).p(y_j|x_m).K_{lm}}{\left(\sum_l p(x_l).p(y_j|x_l)\right)^2}$$

Thus, the value of the distortion between a data point and cluster centroid in the higher dimensional space can be computed using the expression:

$$d(\phi(x_i), y_j) = K_{ii} - 2.\frac{\sum_l p(x_l).p(y_j|x_l).K_{il}}{\sum_l p(x_l).p(y_j|x_l)} + \frac{\sum_{l,m} p(x_l).p(x_m).p(y_j|x_l).p(y_j|x_m).K_{lm}}{\left(\sum_l p(x_l).p(y_j|x_l)\right)^2}$$

$$\forall \ i \in [1, \dots, N], j \in \{1, .., K\}$$

The expression for computing the association probabilities remain the same as in DA algorithm and is given by:

$$p(y_j|x_i) = \frac{\exp\left(-d(\phi(x_i), y_j)/T\right)}{Z_{x_i}}$$

Where, $Z_{x_i} = \sum_{j=1}^{K} \exp\left(-d(\phi(x_i), y_j)/T\right) \forall \ i \in \{1, .., N\}, j \in \{1, .., K\}$

We now have a method for calculating association probabilities and distortion function in the higher-dimensional feature space. The rest of the annealing process remains the same for WKDA algorithm as derived for the DA algorithm.

4.3 PSEUDO CODE FOR THE ALGORITHM

Step 1: Initialise all the input parameters of the algorithm, namely – Number of clusters (K) Initial Temperature (T), Minimum Temperature (T_{min}), Cooling rate (α), Convergence margin ($error$), standard deviation (σ).

Step 2: Let N be the number of data points, M be the dimension of each data point, and K be the number of clusters. Initialise - the distance matrix (size: NxK) with zeros, and the association probability matrix (size: NxK) with random numbers $\in (0,1)$ such that the sum of all the association probabilities in every row is equal to 1. Unless specifically stated, assign $p(x_i) = {}^{1}/_{N} \; \forall$ i ∈ {1, … , N}.

Step 3: Calculate the Gaussian kernel matrix (size: NxN) using the formula:

$$k(x_i, x_j) = \exp\left(\frac{-d(x_i, x_j)}{2.\sigma^2}\right) \forall \text{ i, j} \in \{1, \dots, \text{N}\}$$

$$where, distortion \;\; d(x_i, x_j) = \sum_{q=1}^{M} \left(x_i^q - x_j^q\right)^2.$$

Step 4: Run the loop given below:

While ($T > T_{min}$) {

While (true) {

Step 4(a): Calculate all distance matrix using

$$d(\phi(x_i), y_j) = K_{ii} - 2.\frac{\sum_l p(x_l).p(y_j|x_l).K_{il}}{\sum_l p(x_l).p(y_j|x_l)} + \frac{\sum_{l,m} p(x_l).p(x_m).p(y_j|x_l).p(y_j|x_m).K_{lm}}{\left(\sum_l p(x_l).p(y_j|x_l)\right)^2}$$

$$\forall \text{ i} \in [1, \dots, \text{N}] \text{ , j} \in \{1, .., \text{K}\}$$

Step 4(b): Calculate the new values of association probabilities using

$$p(y_j|x_i) = \frac{\exp\left({}^{-d(\phi(x_i), y_j)}/_{T}\right)}{Z_{x_i}}$$

Where, $Z_{x_i} = \sum_{j=1}^{K} \exp\left({}^{-d(\phi(x_i), y_j)}/_{T}\right) \forall$ i ∈ {1, … , N} , j ∈ {1, . . , K}

If (Converged) break;

}

$T = T * \alpha$; /* the cooling process */

}

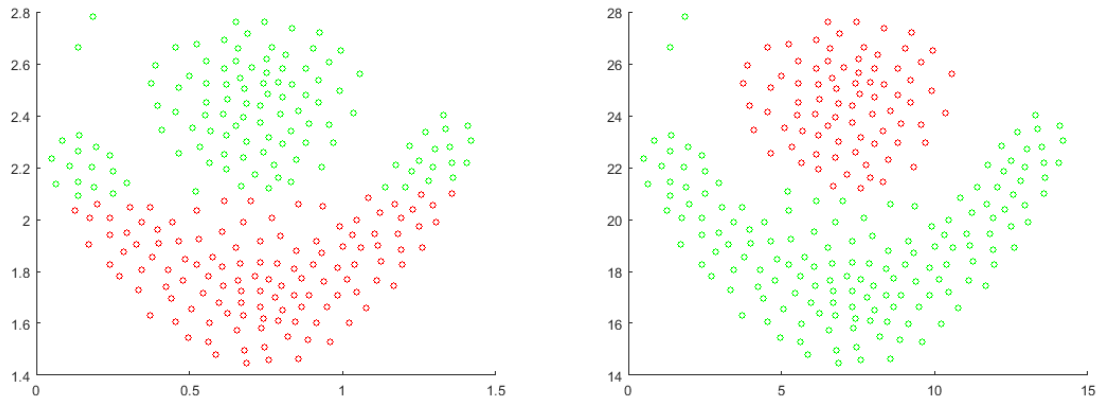
Step 5: Assign clusters to data points using the association probability matrix.

4.4 ANALYSIS OF THE ALGORITHM

The WKDA algorithm works similar to DA algorithm except that we do not explicitly compute the code vectors here, and alternate between calculating the distance matrix and association probability matrix in the central iterations. The most computationally intensive step during each iteration is the calculation of the distance matrix $[d(\phi(x_i), y_j)]$. The worst case time complexity of this step is $O(N^4K)$. Here, N is total number of input data point and K is the number of desired clusters. Thus, if r is the total number of iterations for which the algorithm runs, then the worst case time complexity of the WKDA algorithm is $O(N^4.K.r)$. Similar to the DA algorithm, there are several input parameters for WKDA namely - $T_{initial}, T_{min}$, α, $error$ and σ that one must carefully select according to the input data to run the algorithm successfully.

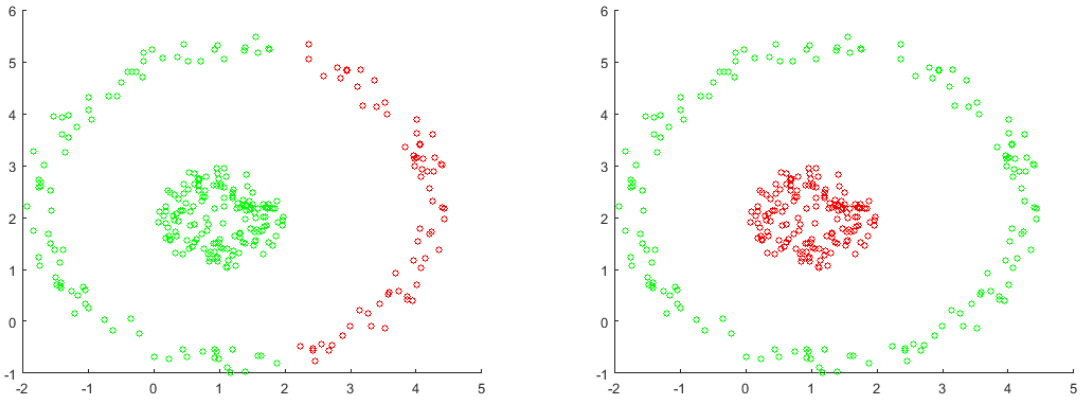
A slight modification to the WKDA algorithm enables us to implement the must-link (the selected data points must be in the same cluster) or cannot-link (the selected data points cannot be in the same cluster) constraints between selected data points. For every cannot-link constraint between x_i and $x_{i'}$, the corresponding entries in the kernel matrix should manually set to zero, i.e. $K(i, i') = 0$. For every must-link constraint between$x_i$and$x_{i'}$, we must manually enforce that their association probabilities are equal, i.e. $p(y_j|x_i) = p(y_j|x_{i'})$ during each outer iteration of the WKDA algorithm.

The WKDA algorithm is tested on several test cases and the results compared with the results of the DA algorithm are given below:



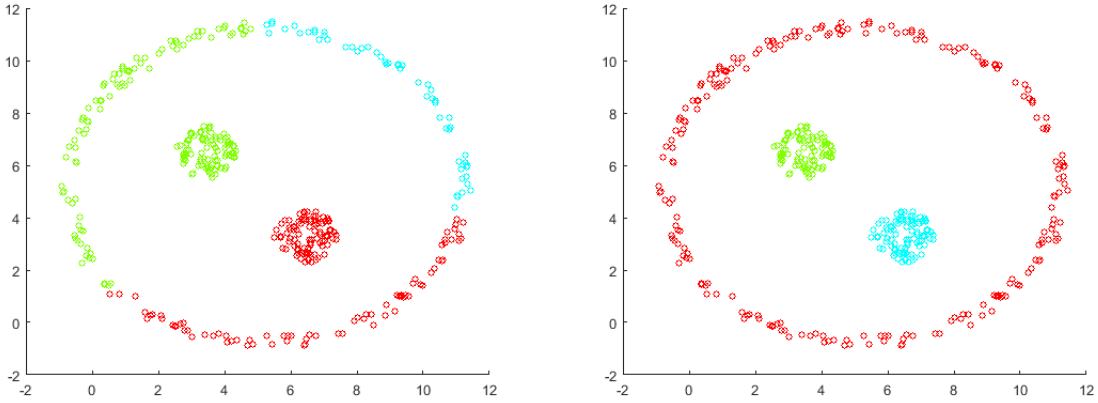
(a) Clustering by DA algorithm *(b) Clustering by WKDA algorithm*

Figure 7 – Result of clustering on data set (N=240, M=2, K=2) by different algorithms



(a) Clustering by DA algorithm *(b) Clustering by WKDA algorithm*

Figure 8 – Result of clustering on data set (N=300, M=2, K=2) by different algorithms



(a) Clustering by DA algorithm *(b) Clustering by WKDA algorithm*

Figure 9 – Result of clustering on data set (N=400, M=2, K=3) by different algorithms

As it can be observed from the figure 7(a), 8(a), and 9(a), while the DA algorithm fails to partition the input space in the desired way, the WKDA algorithm successfully partitions the space with non-linear separators as in figure 7(b), 8(b), and 9(b).

The WKDA algorithm overcomes both the limitations posed by the simpler clustering algorithms. Although it is computationally expensive, but it requires only one initialisation and is able to partition the data with non-linear separators.

CHAPTER 5
CONCLUSION

In this thesis, we first introduce the idea of clustering analysis which is a type of unsupervised learning method used to partition data with the help of an example. Then, we go on to explain the simplest algorithm used for clustering, namely, the k-means algorithm. The main idea of this algorithm is to, initially, randomly define K centroids, one for each cluster, and then iteratively improve the location of the centroids so as to reach a local optimum. Although intuitive and computational efficient, the algorithm has several drawbacks, such as the curse of initialisation and its inability to partition data that is not linearly separable in the desired way, thus limiting us from using this algorithm in complex cases. This motivates us to look into other approaches used for clustering.

We then look into deterministic annealing algorithm, generally used for non-convex optimisation in several fields. The deterministic annealing approach to annealing is slightly computationally expansive over the k-means method but demonstrates substantial performance improvement over it. It has the ability to avoid shallow local optima of the cost function and has the ability to reach the global optimum in a single run of the code. It is derived within a probabilistic framework from basic information theoretic principles, such as, maximum entropy principle, and has the ability to reach the solutions that are totally independent of the choice of initial configurations. Although the DA algorithm overcomes the curse of initialisation, it can't separate data that is not linearly separable in the input space.

We finally look into the innovative WKDA algorithm for shape clustering. The algorithm combines the kernel trick with distributed aspect of the deterministic annealing algorithm to produce effective clustering solutions that are independent of initialisation and has the ability to partition data that is not linearly separable in the input space in the desired way. We test the WKDA algorithm on several non-linear shape clustering scenarios and obtain the desired results.

In the future, we would like to try using the concept of Tsallis entropy instead of Shannon entropy in the WKDA algorithm to see if it results in any performance improvements. Also, we would like to optimise the WKDA algorithm further so that we can work with faster annealing schedules and thus reducing its worst case time complexity.

REFERENCES

[1] MacQueen, J. Some methods of classification and analysis of multivariate observations Cam, Le, L. M. Neyman J. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability 1 281–297 Berkeley, CA: University of California Press 1967.

[2] S. P. Lloyd, "Least squares quantization in pcm," Information Theory, IEEE Transactions on, vol. 28, no. 2, pp. 129–137, 1982.

[3] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004, pp. 551–55.

[4] Rose, Kenneth. "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems." Proceedings of the IEEE 86.11 (1998): 2210-2239.

[5] Pratik M. Parekh, Dimitrios Katselis, Carolyn L. Beck, Srinivasa M. Salapaka. "Deterministic Annealing for Clustering: Tutorial and Computational Aspects". 2015 American Control Conference, July 1-3, 2015. Chicago, IL, USA.

[6] Jaynes, Edwin T. "Information theory and statistical mechanics." Physical review 106.4 (1957): 620.

[7] D. Bertsimas, J. Tsitsiklis, "Simulated Annealing", Statistical Science, vol. 8, no. 1, pp. 10-15, 1993.

[8] Banarwal, Mayank. "Weighted Kernel Deterministic Annealing: A Maximum-Entropy Principle Approach for Shape Clustering". Indian Control Conference, IIT Kanpur, India, January 2018.

[9] W. H. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," Journal of classification, vol. 1, no. 1, pp. 7–24, 1984.

[10] N. UDEA, "Deterministic annealing variant of the EM algorithm", Advances in Neural Information Processing Systems 7 (NIPS7), pp. 69–77, 1995.

[11] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: a kernel approach," in Proceedings of the 22nd international conference on Machine learning. ACM, 2005, pp. 457– 464.

[12] P. Sharma, S. Salapaka, and C. Beck, “A scalable deterministic annealing algorithm for resource allocation problems,” in American Control Conference, 2006. IEEE, 2006, pp. 6–pp.

[13] B. Hajek, “Cooling Schedules for Optimal Annealing", Math. Oper. Res., (1988)13: 311-329.

[14] P. V. Gehler and O. Chapelle, “Deterministic annealing for multiple instance learning.” in AIStats, 2007, pp. 123–130.