Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 00 (2010) 1–10
Procedia Computer Science

International Conference on Computational Science, ICCS 2010

Generative topographic mapping by deterministic annealing

Jong Youl Choi[a,b], Judy Qiu[a], Marlon Pierce[a], Geoffrey Fox[a,b]

[a] *Pervasive Technology Institute, Indiana University, Bloomington IN, USA*
[b] *School of Informatics and Computing, Indiana University, Bloomington IN, USA*

Abstract

Generative Topographic Mapping (GTM) is an important technique for dimension reduction which has been successfully applied to many fields. However the usual Expectation-Maximization (EM) approach to GTM can easily get stuck in local minima and so we introduce a Deterministic Annealing (DA) approach to GTM which is more robust and less sensitive to initial conditions so we do not need to use many initial values to find good solutions. DA has been very successful in clustering, hidden Markov Models and Multidimensional Scaling but typically uses a fixed cooling schemes to control the temperature of the system. We propose a new cooling scheme which can adaptively adjust the choice of temperature in the middle of process to find better solutions. Our experimental measurements suggest that deterministic annealing improves the quality of GTM solutions.

Keywords: deterministic annealing, Generative Topographic Mapping, nonlinear optimization
2010 MSC: 65K10, 65J15

1. Introduction

Visualization of high-dimensional data in a low-dimension space is the core of exploratory data analysis in which users seek the most meaningful information hidden under the intrinsic complexity of data mostly due to high dimensionality. Among many tools available thanks to the recent advances in machine learning techniques and statistical analysis, Generative Topographic Mapping (GTM) has been extensively studied and successfully used in many application areas: biology [1, 2], medicine [3, 4], feature selection [5], to name a few.

Generative Topographic Mapping (GTM) [6, 7] is an unsupervised learning method developed for modeling the probability density of data and finding a non-linear mapping of high-dimensional data onto a low-dimensional space. Similar algorithms known for dimension reduction are Principle Component Analysis (PCA), Multidimensional Scaling (MDS) [8], and Self-Organizing Map (SOM) [9]. In contrast to the Self-Organizing Map (SOM) which does not have any density model [7], GTM defines an explicit probability density model based on Gaussian distribution. For this reason, GTM is also known as a principled alternative to SOM.

The problem challenged by the GTM is to find the best set of parameters associated with Gaussian mixtures by using an optimization method, notably the Expectation-Maximization (EM) algorithm [6, 7]. Although the EM algorithm [10] has been widely used in many optimization problems, it has been shown a severe limitation, known

Email addresses: jychoi@cs.indiana.edu (Jong Youl Choi), xqiu@indiana.edu (Judy Qiu), mpierce@cs.indiana.edu (Marlon Pierce), gcf@cs.indiana.edu (Geoffrey Fox)

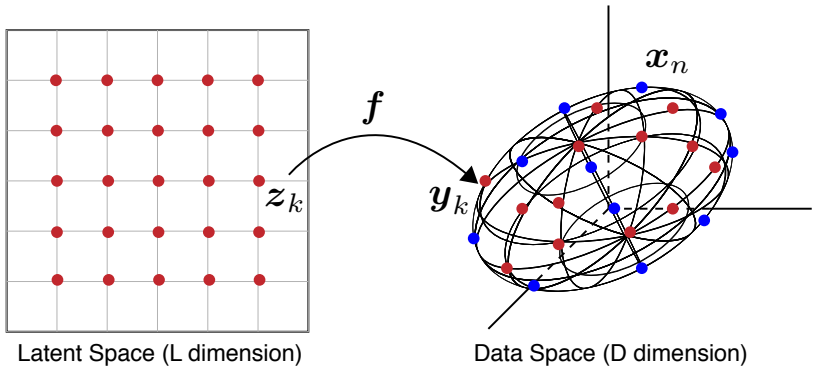


Figure 1: Non-linear embedding by GTM

as a local optima problem [11], in which the EM method can be easily trapped in local optima failing to return global optimum and so the outputs are very sensitive to initial conditions. To overcome such problem occurred in the original GTM [6, 7], we have applied a robust optimization method known as Deterministic Annealing (DA) [12] to seek global solutions, not local optima. Up to our best knowledge, this is the first work to use DA algorithm to solve the GTM problem in the literature.

The DA algorithm [12, 13, 14, 15] has been successfully applied to solve many optimization problems in various machine learning algorithms and applied in many problems, such as clustering [16, 12, 17], visualization [18], protein alignment [19], and so on. The core of the DA algorithm is to seek an global optimal solution in a deterministic way [12], which contrasts to stochastic methods used in the simulated annealing [20], by controlling the level of randomness. This process, adapted from a physical process, is known as annealing in that an optimal solution is gradually revealing as lowering *temperature* which controls randomness. At each level of temperature , the DA algorithm chooses an optimal solution by using the principle of maximum entropy [21], a rational approach to choose the most unbiased and non-committal answers for given conditions.

Regarding the cooling process, which affects the speed of convergency, in general it should be slow enough to get the optimal solutions. Two types of fixed cooling schemes, exponential cooling schemes in which temperature T is reduced exponentially such that $T = \alpha T$ with a coefficient $\alpha < 1$ and linear scheme like $T = T - \delta$ for $\delta > 0$, are commonly used. Choosing the best cooling coefficient α or δ is very problem dependent and no common rules have been reported. Beside of those conventional schemes, in this paper we also present a new cooling scheme, named *adaptive cooling schedule*, which can improve the performance of DA by dynamically computing the next temperatures during the annealing process with no use of fixed and predefined ones.

The main contributions of our paper are as follow:

i. Developing the DA-GTM algorithm which uses the DA algorithm to solve GTM problem. Our DA-GTM algorithm can give more robust answers than the original EM-based GTM algorithm, not suffering from the local optima problem (Section 3).
ii. Deriving closed-form equations to predict phase transitions which is a characteristic behavior of DA [14]. Our phase transition formula can give a guide to set the initial temperature of the DA-GTM algorithm (Section 4).
iii. Developing an adaptive cooling schedule scheme, in which the DA-GTM algorithm can automatically adjust the cooling schedule of DA in an on-line manner. With this scheme, users are not required to set parameters related with cooling schedule in DA (Section 5).

Our experiment results showing the performance of DA-GTM algorithm in real-life application will be shown in Section 6 followed by our conclusion in Section 7.

2. Overview of GTM

We start by briefly reviewing the original GTM algorithm [6, 7]. The GTM algorithm is to find a non-linear manifold embedding of K latent variables $\boldsymbol{z}_k$ in low L-dimension space or latent space, such that $\boldsymbol{z}_k \in \mathbb{R}^L (k = 1, ..., K)$, which can optimally represent the given N data points $\boldsymbol{x}_n \in \mathbb{R}^D (n = 1, ..., N)$ in the higher D-dimension space or data space (usually $L \ll D$) (Figure 1). This is achieved by two steps: First, mapping the latent variables in the latent space to the data space with respect to a non-linear mapping $f : \mathbb{R}^L \mapsto \mathbb{R}^D$. Let us denote the mapped points in the data space as $\boldsymbol{y}_k$. Secondly, estimating probability density between the mapped points $\boldsymbol{y}_k$ and the data points $\boldsymbol{x}_n$ by using the Gaussian noise model in which the distribution is defined as an isotropic Gaussian centered on $\boldsymbol{y}_k$ having variance β^{-1} (β is known as precision). More specifically, the probability density $p(\boldsymbol{x}_n|\boldsymbol{y}_k)$ is defined by the following Gaussian distribution:

$$\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{y}_k, \beta) = \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}||\boldsymbol{x}_n - \boldsymbol{y}_k||^2\right\}. \tag{1}$$

The choice of the non-linear mapping $f : \mathbb{R}^L \mapsto \mathbb{R}^D$ can be made from any parametric, non-linear model. In the original GTM algorithm [6, 7], a generalized linear regression model has been used, in which the map is a linear combination of a set of fixed M basis functions, such that,

$$\boldsymbol{y}_k = \boldsymbol{\phi}_k^{\mathsf{Tr}} \boldsymbol{W}, \tag{2}$$

where a column vector $\boldsymbol{\phi}_k = (\phi_{k1}, ..., \phi_{kM})$ is a mapping of z_k by the M basis function $\phi_m : \mathbb{R}^L \mapsto \mathbb{R}$ for $m = 1, ..., M$, such that $\phi_{km} = \phi_m(z_k)$ and $\boldsymbol{W}$ is a $M \times D$ matrix containing weight parameters. $\boldsymbol{A}^{\mathsf{Tr}}$ represents a transpose of matrix $\boldsymbol{A}$.

With this model setting, the objective of GTM algorithm corresponds to a Gaussian mixture problem to find an optimal set of $\boldsymbol{y}_k$'s which maximizes the following log-likelihood:

$$\mathcal{L}(\boldsymbol{W}, \beta) = \underset{\boldsymbol{W},\beta}{\operatorname{argmax}} \sum_{n=1}^{N} \ln \left\{ \frac{1}{K} \sum_{k=1}^{K} p(\boldsymbol{x}_n|\boldsymbol{y}_k) \right\} \tag{3}$$

It is the problem that finding K centers for N data points, known as K-clustering problem. Since K-clustering problem is NP-hard [22], the GTM algorithm uses the EM method which starts with initial random weight matrix $\boldsymbol{W}$ and iteratively refines it to maximize Eq. (3), which can be easily trapped in local optima. More details can be found in the original GTM paper [6, 7].

3. GTM with Deterministic Annealing (DA-GTM)

Since the use of EM, the original GTM algorithm suffers from the local optima problem in which the GTM map can vary depending on the initial parameters. Instead of using the EM, we have applied the DA approach to find a global optimum solution. With the help of DA algorithm, we can have more robust GTM maps against the random initial value problem.

In fact, the GTM algorithm's objective function (3) is exactly the same problem, so called K-clustering, discussed by K. Rose in [12]. The problem is to seek optimal clusters for a given distance or distortion measurement by using the deterministic annealing approach. Thus, we can use the same approach used in the Rose's DA method. However, in Rose's paper, squared Euclidean distance has been used in distortion measurement but in the GTM algorithm, distances are measured by the Gaussian probability as in Eq. (1). Thus, we need to redefine distortion between two points as $d(\boldsymbol{x}_n, \boldsymbol{y}_k) = -\log(p(\boldsymbol{x}_n|\boldsymbol{y}_k)/T)$, which leads us to the so-called partitioning function as follows:

$$Z_n = \sum_{k=1}^{K} \exp\left(\frac{-d(\boldsymbol{x}_n, \boldsymbol{y}_k)}{T}\right). \tag{4}$$

In the DA algorithm, we have a new objective function, known as a *free energy*, which we want to minimize through iterations. Now, by using the same definition of free energy, used in Rose's paper in [12], we can drive a *free*

energy function, for the DA-GTM algorithm as follows:

$$F(\boldsymbol{W},\beta,T) = \underset{\boldsymbol{W},\beta,T}{\operatorname{argmin}} \; -T\sum_{n=1}^{N}\ln Z_n \tag{5}$$

$$= \underset{\boldsymbol{W},\beta,T}{\operatorname{argmin}} \; -T\sum_{n=1}^{N}\ln\left\{\left(\frac{1}{K}\right)^{\frac{1}{T}}\sum_{k=1}^{K}p(\boldsymbol{x}_n|\boldsymbol{y}_k)^{\frac{1}{T}}\right\} \tag{6}$$

which we want to minimize as lowering temperature such that $T \to 1$.

Notice that the GTM algorithm's objective function (3) differs only the use of temperature T with our function $F(\boldsymbol{W},\sigma,T)$. Especially, at $T = 1$, $\mathcal{L}(\boldsymbol{W},\beta) = -F(\boldsymbol{W},\beta,T)$ and so the original GTM algorithm's target function can be considered as a special case of the DA-GTM algorithm's.

To minimize (6), we need to make the following two partial derivatives be zero:

$$\frac{\partial F}{\partial \boldsymbol{y}_k} = \beta\sum_{n=1}^{N}\rho_{kn}(\boldsymbol{x}_n - \boldsymbol{y}_k) \tag{7}$$

$$\frac{\partial F}{\partial \beta} = \sum_{n=1}^{N}\sum_{k=1}^{K}\rho_{kn}\left(\frac{D}{2\beta} - \frac{1}{2}\|\boldsymbol{x}_n - \boldsymbol{y}_k\|^2\right) \tag{8}$$

where ρ_{kn} is a property, known as *responsibility*, such that,

$$\rho_{kn} = \frac{p(\boldsymbol{x}_n|\boldsymbol{y}_k)^{\frac{1}{T}}}{\sum_{k'=1}^{K}p(\boldsymbol{x}_n|\boldsymbol{y}_{k'})^{\frac{1}{T}}} \tag{9}$$

By using the same matrix notations used in the GTM paper [6, 7], the DA-GTM algorithm can be written as a process to seek an optimal weights $\boldsymbol{W}$ and precision β at each temperature T.

$$\boldsymbol{W} = (\boldsymbol{\Phi}^{\mathsf{Tr}}\boldsymbol{G}'\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathsf{Tr}}\boldsymbol{R}'\boldsymbol{X} \tag{10}$$

$$\beta = \frac{1}{ND}\sum_{n}^{N}\sum_{k}^{K}\rho_{kn}\|\boldsymbol{x}_n - \boldsymbol{y}_k\|^2 \tag{11}$$

where $\boldsymbol{X}$ is a $N\times D$ data matrix, $\boldsymbol{\Phi}$ is a $K\times M$ basis matrix, $\boldsymbol{G}'$ is a $K\times K$ diagonal matrix with elements $g'_{kk} = \sum_n^N(\rho_{kn})^{\frac{1}{T}}$.

4. Phase Transitions of DA-GTM

As a characteristic behavior of the DA algorithm explained by Rose in [12], it undergoes phase transitions as lowering the temperatures. At some temperature, we can not obtain all solutions but, instead, we can only obtain effective number of solutions. All solutions will gradually pop out while the annealing process proceeds as with lowering the temperature.

In the DA-GTM algorithm, we can observe the same behavior. As an extreme example, at very high temperature, the DA-GTM algorithm gives only one effective latent point in which all $\boldsymbol{y}_k$'s are converged into the same point $\bar{\boldsymbol{x}}$ which is the center of data points, such that $\bar{\boldsymbol{x}} = \sum_{n=1}^{N}\boldsymbol{x}_n/N$. As lowering the temperature under a certain point, $\boldsymbol{y}_k$'s settled in the same position start to *"explode"*. We call this temperature as the first critical temperature, denoted by T_c. As we further lowering temperature, we can observe subsequent phase transitions and so existence of multiple critical temperatures. Computing the first phase transition is an important task since we should start our annealing process with the initial temperature larger than T_c.

In DA, we can define such phase transitions as a moment of loosing stability of the objective function, the free energy F, and turning to be unstable. Mathematically, that moment corresponds to the point in which the Hessian of the object function looses its positive definiteness.

For our DA-GTM algorithm, we can write the following Hessian matrix as a block matrix:

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1K} \\ \vdots & & \vdots \\ H_{K1} & \cdots & H_{KK} \end{bmatrix}, \tag{12}$$

where a sub matrix H_{ij} is a second derivative of the free energy F as shown in Eq. (6). More specifically, H_{ij} can be written as follows:

$$H_{kk} = \frac{\partial^2 F}{\partial \boldsymbol{y}_k \partial \boldsymbol{y}_k^{\mathsf{Tr}}} = -\sum_n^N \left\{ \frac{\beta^2}{T} \rho_{kn}(1-\rho_{kn})(\boldsymbol{x}_n - \boldsymbol{y}_k)^{\mathsf{Tr}}(\boldsymbol{x}_n - \boldsymbol{y}_k) - \beta \rho_{kn} \boldsymbol{I}_D \right\}, \text{ or} \tag{13}$$

$$H_{kk'} = \frac{\partial^2 F}{\partial \boldsymbol{y}_k \partial \boldsymbol{y}_{k'}^{\mathsf{Tr}}} = \sum_n^N \left\{ \frac{\beta^2}{T} \rho_{kn}\rho_{k'n}(\boldsymbol{x}_n - \boldsymbol{y}_k)^{\mathsf{Tr}}(\boldsymbol{x}_n - \boldsymbol{y}_{k'}) \right\} (k \neq k'), \tag{14}$$

where $k, k' = 1, \ldots, K$, and $\boldsymbol{I}_D$ is an identity matrix of size D. Note that H_{kk} and $H_{kk'}$ are $D \times D$ matrices and thus, $H \in \mathbb{R}^{KD \times KD}$.

With the Hessian matrix above, we can compute the first phase transition point occurred at T_c. Assuming that the system hasn't undergone the first phase transition and the current temperature is high enough, then we will have all $\boldsymbol{y}_k$'s settled in the center of the data point, denoted by $\boldsymbol{y}_0 = \bar{\boldsymbol{x}} = \sum_{n=1}^N \boldsymbol{x}_n / N$, and equal responsibilities as follows:

$$\boldsymbol{y}_k = \boldsymbol{y}_0 \text{ and } \rho_{kn} = 1/K \tag{15}$$

for all $k = 1, ..., K$. Then, the second derivatives can be rewritten by

$$H_{kk} = -\frac{\beta^2 N}{TK^2} \left\{ (K-1)\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} - \frac{TK}{\beta} \boldsymbol{I}_D \right\} \tag{16}$$

$$H_{kk'} = \frac{\beta^2 N}{TK^2} \boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \tag{17}$$

where $\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0}$ represents a covariance matrix of centered data set such that,

$$\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} = \frac{1}{N} \sum_{n=1}^N (\boldsymbol{x}_n - \boldsymbol{y}_0)^{\mathsf{Tr}}(\boldsymbol{x}_n - \boldsymbol{y}_0) \tag{18}$$

and the Hessian matrix also can be rewritten by

$$H = \frac{-\beta^2 N}{TK^2} \left\{ \begin{bmatrix} (K-1)\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} & \cdots & -\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \\ \vdots & \ddots & \vdots \\ -\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} & \cdots & (K-1)\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \end{bmatrix} - \frac{TK}{\beta} \boldsymbol{I}_{KD} \right\} \tag{19}$$

The first phase transition occurs when the system is getting unstable so that the above Hessian matrix is loosing its positive definiteness. I.e., the first phase transition is the moment when the Hessian matrix becomes singular and so its determinant equals 0(zero), such that $\det(H) = 0$ at $T = T_c$, which holds the following:

$$\text{eig}\left(\begin{bmatrix} (K-1)\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} & \cdots & -\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \\ \vdots & \ddots & \vdots \\ -\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} & \cdots & (K-1)\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \end{bmatrix} \right) = \frac{T_c K}{\beta} \tag{20}$$

where $\text{eig}(\boldsymbol{A})$ is an eigenvalue of $\boldsymbol{A}$.

We can further simplify the above equation by using the Kronecker product:

$$\text{eig}\left(\begin{bmatrix} K-1 & \cdots & -1 \\ \vdots & \ddots & \vdots \\ -1 & \cdots & K-1 \end{bmatrix} \otimes \boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \right) = \text{eig}\left(\begin{bmatrix} K-1 & \cdots & -1 \\ \vdots & \ddots & \vdots \\ -1 & \cdots & K-1 \end{bmatrix} \right) \otimes \text{eig}\left(\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0} \right) \tag{21}$$

Since the first critical temperature is the most largest one, we can use only the maximum eigenvalue among multiple of them. Thus, the first critical temperature can be obtained by the following equation:

$$T_c = \beta\lambda_{\max} \tag{22}$$

where $\lambda_{\max}$ is the largest value of $\mathrm{eig}(\boldsymbol{S}_{\boldsymbol{x}|\boldsymbol{y}_0})$. Computing the subsequent critical temperature will be discuss in the next.

5. Adaptive cooling schedule

The DA algorithm has been applied in many areas and proved its success in finding a global optimal solution avoiding local minima. However, up to our knowledge, no literature has been found about the cooling schedule of DA algorithm. Commonly used cooling schedule is exponential, such as $T = \alpha T$, or linear, such as $T = T - \delta$ for some fixed coefficient α and δ. Those scheduling schemes are fixed in that cooling temperatures are pre-defined and the coefficient α or δ will not be changed during the process, regardless of the complexity of a given problem.

However, as we discussed previously, the DA algorithm undergoes the phase transitions in which the solution space can change dramatically. To avoid such drastic changes and make the transitions smoother, one may try to use very small α or δ coefficient. However, the procedure can go too long to be used in practice but still there is no guarantee to produce global optimal solutions (You will see an example in our experiment result in the next).

To overcome such problem, we propose an adaptive cooling schedule in which next cooling temperatures are determined dynamically during the annealing process. More specifically, at every iteration of DA algorithm, we predict the next phase transition temperature and move to the point as quickly as possible.

General approach to find next critical temperatures T_c at any given temperature T is the same as the previous one in that we need to find T_c to make $\det(H) = 0$. However, in contrast to the method used in computing the first critical points, for this problem we can't have any simplified closed-form equations in general. Instead, we can try to divide the problem into pieces to derive closed-form equations and choose the best solution from multiple candidate answers. This method could not provide an exact solution, compared to the method to solve the problem in a whole set, but rather give us an approximation. However, our experiment has showed that such approximation can be a good solution. Also, another advantage we can expect in using our adaptive cooling schedule is that users has no need to set any coefficient for cooling schedule. Cooling process is automatically adjusted to the problem.

At a given state in GTM algorithm, we find K soft clusters in which each group is represented by $\boldsymbol{y}_k$. Instead of finding a global critical temperature from K clusters at once, we can find each critical temperature at each group and choose one among K candidates which is the most largest yet lower than current temperature so that it can be used as a next temperature.

The sketch of the algorithm is as follows: i) For each k cluster ($k = 1, \ldots, K$), find a candidate of next critical temperature $T_{c,k}$ which should satisfy $\det(H_{kk}) = 0$ as defined in (13). ii) Choose the most largest yet lower than the current T among $\{T_{c,k}\}$.

To find $\det(H_{kk}) = 0$, we need to rewrite Eq. (13) as follows:

$$H_{kk} = \frac{-\beta^2}{T}\left(\boldsymbol{U}_{\boldsymbol{x}|\boldsymbol{y}_k} - \boldsymbol{V}_{\boldsymbol{x}|\boldsymbol{y}_k} - \frac{T g'_{kk}}{\beta}\boldsymbol{I}_D\right) \tag{23}$$

with the following definition:

$$\boldsymbol{U}_{\boldsymbol{x}|\boldsymbol{y}_k} = \sum_{n=1}^{N} \rho_{kn}(\boldsymbol{x}_n - \boldsymbol{y}_k)^{\mathsf{Tr}}(\boldsymbol{x}_n - \boldsymbol{y}_k) \tag{24}$$

$$\boldsymbol{V}_{\boldsymbol{x}|\boldsymbol{y}_k} = \sum_{n=1}^{N} (\rho_{kn})^2(\boldsymbol{x}_n - \boldsymbol{y}_k)^{\mathsf{Tr}}(\boldsymbol{x}_n - \boldsymbol{y}_k) \tag{25}$$

and $g'_{kk} = \sum_{n=1}^{N} \rho_{kn}$. Then, the condition $\det(H_{kk}) = 0$ at $T = T_{c,k}$ will hold $\mathrm{eig}\left(\boldsymbol{U}_{\boldsymbol{x}|\boldsymbol{y}_k} - \boldsymbol{V}_{\boldsymbol{x}|\boldsymbol{y}_k}\right) = \frac{g'_{kk}}{\beta}T_{c,k}$ and thus the next critical temperature $T_{c,k}$ for the cluster k can computed by

$$T_{c,k} = \frac{\beta}{g'_{kk}}\lambda_{\max,k} \tag{26}$$

Algorithm 1 GTM with Deterministic Annealing

DA-GTM

Set $T > T_c$ by using Eq. (22)
Choose randomly M basis function $\phi_m (m = 1, ..., M)$
Compute $\mathbf{\Phi}$ whose element $\phi_{km} = \phi_m(\boldsymbol{z}_k)$
Initialize randomly $\boldsymbol{W}$
Compute β by Eq. (11)
while $T \geq 1$ **do**
  Update $\boldsymbol{W}$ by Eq. (10)
  Update β by Eq. (11)
  $T \leftarrow$ NextCriticalTemp
end while
return $\mathbf{\Phi}, \boldsymbol{W}, \beta$

Algorithm 2 Find the next critical temperature

NextCriticalTemp

for $k = 1$ to K **do**
  $\Lambda_k \leftarrow \{\emptyset\}$
  for each $\lambda \in \mathrm{eig}\left(\boldsymbol{U}_{\boldsymbol{x}|\boldsymbol{y}_k} - \boldsymbol{V}_{\boldsymbol{x}|\boldsymbol{y}_k}\right)$ **do**
    if $\lambda < T g'_{kk}/\beta$ **then**
      $\Lambda_k \leftarrow \Lambda_k \cup \lambda$
    end if
  end for
  $\lambda_{\max,k} \leftarrow \max(\Lambda_k)$
  $T_{c,k} \leftarrow \beta \lambda_{\max,k}/g'_{kk}$
end for
return $T_c \leftarrow \max(\{T_{c,k}\})$

where $\lambda_{\max,k}$ is the largest but lower than $T g'_{kk}/\beta$ eigenvalue of the matrix $\left(\boldsymbol{U}_{\boldsymbol{x}|\boldsymbol{y}_k} - \boldsymbol{V}_{\boldsymbol{x}|\boldsymbol{y}_k}\right)$. The overall pseudo code is shown in Algorithm 1 and Algorithm 2.

6. Experiment Results

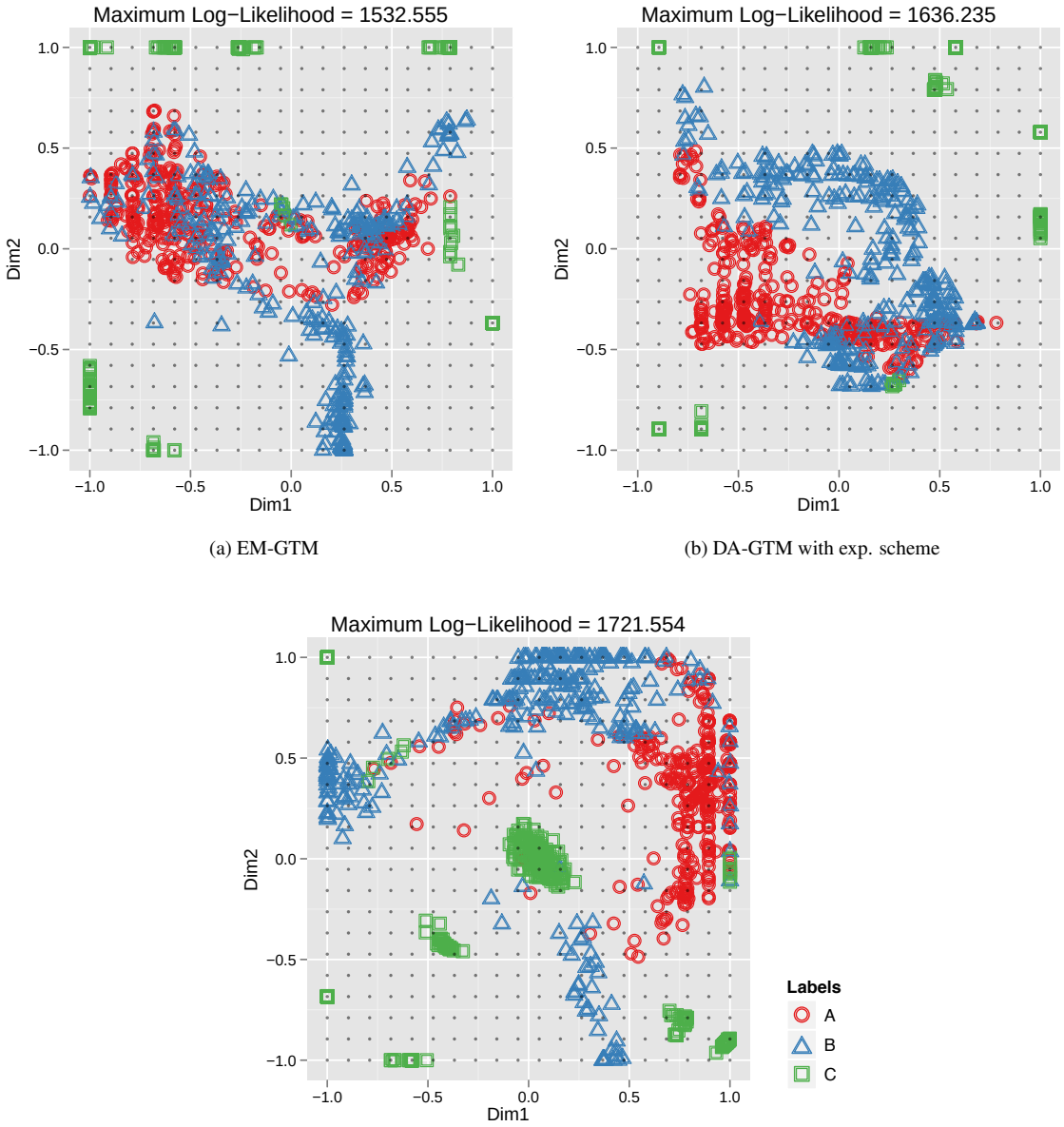
To compare the performances of our DA-GTM algorithm with the original EM-based GTM (EM-GTM hereafter for short), we have performed a set of experiments by using two datasets: i) the oil flow data used in the original GTM paper [6, 7] obtained from the GTM website[1], which has 1,000 points having 12 dimensions for 3-phase clusters and ii) chemical compound dataset obtained from PubChem database[2], which is a NIH-funded repository for over 60 million chemical molecules and provides their chemical structure fingerprints and biological activities, for the purpose of chemical information mining and exploration. In this paper we have randomly selected 1,000 element subset having 166 dimensions.

First we have compared for the oil-flow data maximum log-likelihood produced by EM-GTM, DA-GTM with exponential cooling scheme, and DA-GTM with adaptive cooling scheme and their GTM plots, known as "posterior-mean projection" plot [6, 7], in the latent space. For each algorithm, we have executed 10 runs with random setup, chosen a median result, and drawn a GTM plot as shown in Figure 2. As a result, DA-GTM with adaptive cooling scheme (c) has produced the largest maximum log-likelihood (best performance), while EM-GTM (a) produced the smallest maximum log-likelihood (worst performance). Also, as seen in the figures, a plot with larger maximum log-likelihood shows better separation of the clusters.

In the next, we have compared the performance of EM-GTM and DA-GTM with 3 cooling schedule schemes: i) Adaptive, which we have prosed in this paper, ii) Exponential with cooling coefficients $\alpha = 0.95$ (denoted Exp-A hereafter), and iii) Exponential with cooling coefficients $\alpha = 0.99$ (denoted Exp-B hereafter). For each DA-GTM setting, we have also applied 3 different starting temperature 5, 6, and 7, which are all larger than the 1st critical temperature which is about 4.64 computed by Eq. (22). Figure 3 shows the summary of our experiment results in which numbers are estimated by the average of 50 executions with random initialization.

As a result shown in Figure 3, DA-GTM algorithm shows strong robustness against local minima since the mean of log-likelihood from DA-GTM's outperforms EM-GTM's by about 11.15% even with smaller deviations. Interestingly, at the starting temperature 7 and 9, the performance of Exp-B, which used more finer-grain cooling schedule ($\alpha = 0.99$) than Exp-A ($\alpha = 0.95$), is lower than Exp-A and even under EM-GTM. This shows that fine-grained cooling schedule can not guarantee the best solution than corse-grained one in using DA. However, our adaptive cooling scheme mostly outperforms among other cooling schemes. Figure 4 shows an example of execution of DA-GTM algorithm with adaptive cooling schedule.

[1] GTM homepage, `http://www.ncrg.aston.ac.uk/GTM/`
[2] PubChem project, `http://pubchem.ncbi.nlm.nih.gov/`



(a) EM-GTM

(b) DA-GTM with exp. scheme

(c) DA-GTM with adaptive scheme

Figure 2: Comparison of (a) EM-GTM, (b) DA-GTM with exponential, and (c) DA-GTM with adaptive cooling scheme for the oil-flow data having 3-phase clusters(A=Homogeneous, B=Annular, and C=Stratified configuration). Plots are drawn by a median result among 10 random-initialized executions for each scheme. As a result, DA-GTM (c) with adaptive cooling scheme has produced the largest maximum log-likelihood and the plot shows better separation of the clusters, while EM-GTM (a) has output the smallest maximum log-likelihood and the plot shows many overlaps.

We have also compared EM-GTM and DA-GTM with exponential cooling scheme for the 1,000 element PubChem dataset which has 166 dimensions. As shown in Figure 5, DA-GTM's output is better than EM-GTM's since DA-GTM's average maximum log-likelihood (-36,608) is larger than EM-GTM's (-36,666).

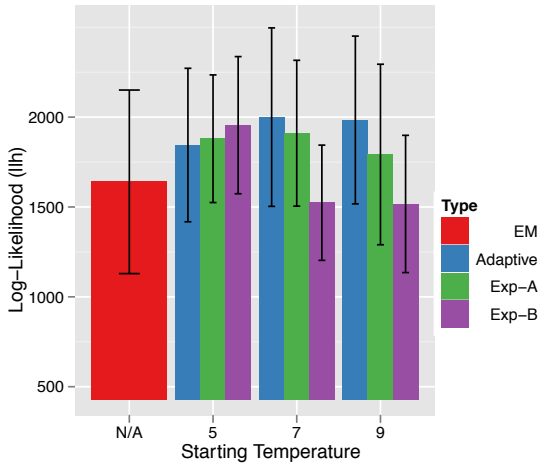
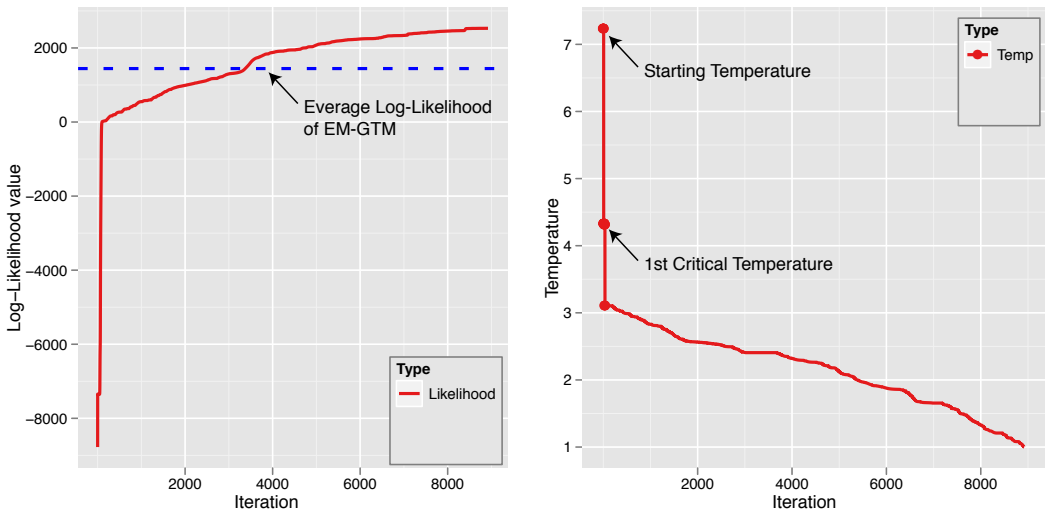


Figure 3: Comparison of EM-GTM with DA-GTM in various settings. Average of 50 random initialized runs are measured for EM-GTM, DA-GTM with 3 cooling schemes (adaptive, exponential with $\alpha = 0.95$ (Exp-A) and $\alpha = 0.99$ (Exp-B).



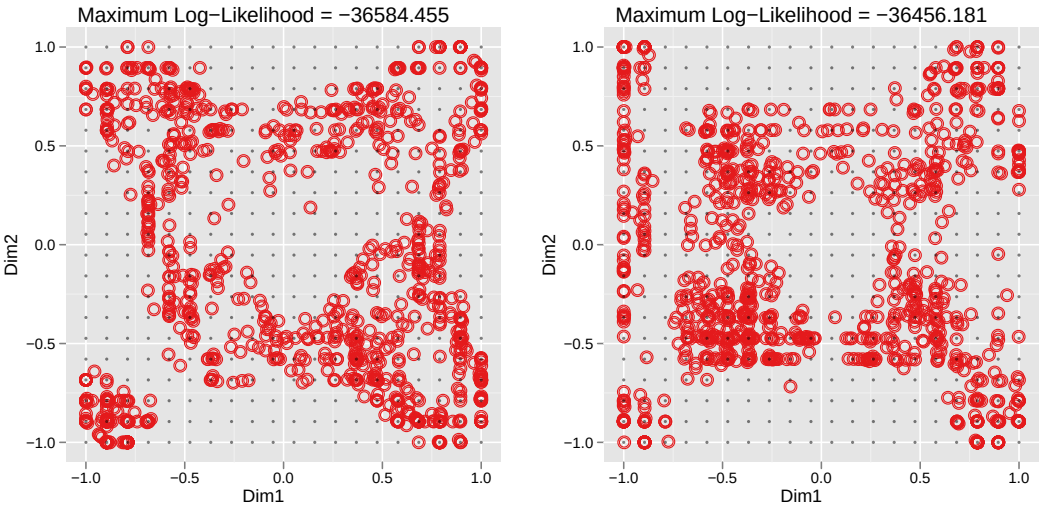
(a) Progress of log-likelihood

(b) Adaptive changes in cooling schedule

Figure 4: A progress of DA-GTM with adaptive cooling schedule. This example show how DA-GTM with adaptive cooling schedule progresses through iterations

7. Conclusion

We have solved the GTM problem, originally using the EM method, by using the deterministic annealing (DA) algorithm which is more resilient against the local optima problem and less sensitive to initial conditions, from which the original EM method was suffered. We have also developed a new cooling scheme, called *adaptive cooling schedule*. In contrast to the conventional cooling schemes such as linear and exponential ones, all of them are pre-defined and fixed, our adaptive cooling scheme can adjust granularity of cooling speed in an on-line manner. In our experiment, our adaptive cooling scheme can outperform other conventional methods.



(a) EM-GTM (b) DA-GTM with exp. scheme

Figure 5: Comparison of (a) EM-GTM and (b) DA-GTM with exponential scheme for 1,000 element PubChem dataset having 166 dimensions. Plots are drawn as a median result of 10 randomly initialized executions of EM-GTM and DA-GTM. The average maximum log-likelihood from DA-GTM algorithm (-36,608) is larger than one from EM-GTM (-36,666).

References

[1] A. Staiano, L. De Vinco, A. Ciaramella, G. Raiconi, R. Tagliaferri, G. Longo, G. Miele, R. Amato, C. Del Mondo, C. Donalek, et al., Probabilistic principal surfaces for yeast gene microarray data-mining, in: Data Mining (ICDM 2004). Proceedings of Fourth IEEE International Conference, 2004, pp. 202–208.

[2] D. D'Alimonte, D. Lowe, I. Nabney, V. Mersinias, C. Smith, MILVA: An interactive tool for the exploration of multidimensional microarray data (2005).

[3] A. Vellido, P. Lisboa, Handling outliers in brain tumour MRS data analysis through robust topographic mapping, Computers in Biology and Medicine 36 (10) (2006) 1049–1063.

[4] D. Maniyar, I. Nabney, Visual data mining using principled projection algorithms and information visualization techniques, in: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2006, p. 648.

[5] A. Vellido, Assessment of an Unsupervised Feature Selection Method for Generative Topographic Mapping, Lecture Notes in Computer Science 4132 (2006) 361.

[6] C. Bishop, M. Svensén, C. Williams, GTM: The generative topographic mapping, Neural computation 10 (1) (1998) 215–234.

[7] C. Bishop, M. Svensén, C. Williams, GTM: A principled alternative to the self-organizing map, Advances in neural information processing systems (1997) 354–360.

[8] J. Kruskal, Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis, Psychometrika 29 (1) (1964) 1–27.

[9] T. Kohonen, The self-organizing map, Neurocomputing 21 (1-3) (1998) 1–6.

[10] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society. Series B (Methodological) 39 (1) (1977) 1–38.

[11] N. Ueda, R. Nakano, Deterministic annealing EM algorithm, Neural Networks 11 (2) (1998) 271–282.

[12] K. Rose, Deterministic annealing for clustering, compression, classification, regression, and related optimization problems, Proceedings of the IEEE 86 (11) (1998) 2210–2239.

[13] K. Rose, E. Gurewitz, G. Fox, A deterministic annealing approach to clustering, Pattern Recognition Letters 11 (9) (1990) 589–594.

[14] K. Rose, E. Gurewitz, G. Fox, Statistical mechanics and phase transitions in clustering, Physical Review Letters 65 (8) (1990) 945–948.

[15] K. Rose, E. Gurewitz, G. Fox, Vector quantization by deterministic annealing, IEEE Transactions on Information Theory 38 (4) (1992) 1249–1257.

[16] T. Hofmann, J. Buhmann, Pairwise data clustering by deterministic annealing, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1) (1997) 1–14.

[17] X. Yang, Q. Song, Y. Wu, A robust deterministic annealing algorithm for data clustering, Data & Knowledge Engineering 62 (1) (2007) 84–100.

[18] H. Klock, J. Buhmann, Multidimensional scaling by deterministic annealing, Lecture Notes in Computer Science 1223 (1997) 245–260.

[19] L. Chen, T. Zhou, Y. Tang, Protein structure alignment by deterministic annealing, Bioinformatics 21 (1) (2005) 51–62.

[20] S. Kirkpatric, C. Gelatt, M. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.

[21] E. Jaynes, On the rationale of maximum-entropy methods, Proceedings of the IEEE 70 (9) (1982) 939–952.

[22] D. Aloise, A. Deshpande, P. Hansen, P. Popat, NP-hardness of Euclidean sum-of-squares clustering, Machine Learning 75 (2) (2009) 245–248.