A Robust and Scalable Solution for Interpolative Multidimensional Scaling With Weighting

Yang Ruan, Geoffrey Fox
School of Informatics and Computer Science
Indiana University
Bloomington, Indiana, USA
{yangruan, gcf}@indiana.edu

***Abstract*— Advances in modern bio-sequencing techniques have led to a proliferation of raw genomic data that enables an unprecedented opportunity for data mining. To analyze such large volume and high-dimensional scientific data, many high performance dimension reduction and clustering algorithms have been developed. Among the known algorithms, we use Multidimensional Scaling (MDS) to reduce the dimension of original data and Pairwise Clustering, and to classify the data. We have shown that an interpolative technique can be applied to get better performance on massive data. However, SMACOF MDS approach is only directly applicable to cases where all pairwise distances are used and where weight is one for each term. In this paper, we proposed a robust and scalable MDS and interpolation algorithm using Deterministic Annealing technique, to solve problems with either missing distances or a non-trivial weight function. We compared our method to three state-of-art techniques. By experimenting on three common types of bioinformatics dataset, the results illustrate that the precision of our algorithms are better than other algorithms, and the weighted solutions has a lower computational time cost as well.**

Keywords—Deterministic Annealing; Multidimensional Scaling

I. Introduction

The speed of data collections by modern instruments in every scientific and technical field is accelerating rapidly with the advancement of science technologies. As this massive amount of data can easily overwhelm a single computer, traditional data analysis technologies needs to be modified and upgraded to adapt to high performance computational environment for acceptable performance. Many data mining and machine learning algorithms have been developed to solve these big data problems. Among those algorithms, dimension reduction has been proved to be useful in data clustering and visualization field [1] [2]. This technique enables the investigation of unknown structures from high dimensional space into visualization in 2D or 3D space.

Multidimensional Scaling (MDS) is one set of techniques among many existing dimension reduction methods, such as Principal Component Analysis (PCA) [3], Generative Topographic Mapping (GTM) [4], and Self-Organizing Maps (SOM) [5]. Different from them, which focus on using the feature vector information in original dimension to construct a configuration in low dimension space, MDS focuses on using the proximity data, which is represented as pairwise dissimilarity values generated from high dimensional space. As in bioinformatics data, one needs to deal with sequences generated from sequencing technology, where the feature vectors are very difficult to be retrieved because of various sequence lengths. It is not suitable to use technologies other than MDS for their dimension reduction.

DACIDR [6] is an application that can generate robust clustering and visualization results on millions of sequences by using MDS technique. In DACIDR, the pairwise dissimilarities can be calculated by pairwise sequence alignment. Then MDS uses the result from it as input. Furthermore, to deal with large-scale data, DACIDR uses an interpolation algorithm called Majorizing Iterative MDS (MI-MDS) [7] to reduce the memory usage. It has been proven that DACIDR could visualize and cluster over millions of sequences with limited computing power. But in our recent study, we found out that pairwise sequence alignment could generate very low quality dissimilarity values in some cases, where these values could cause inaccuracy in clustering and visualization. So in MDS or interpolation, these values should be considered missing. Therefore, in this paper, we propose a robust solution for input data with missing values by adding a weight function to both MDS and interpolation. And we have reduced the time complexity of weighted MDS from cubic to quadratic so that its processing capability could be scaled up. Furthermore, as the MI-MDS uses iterative majorization to solve the non-linear problem of interpolation, it could suffer from local optima problem [8]. So we apply a robust optimization method called Deterministic Annealing [9] [10] (DA) in order to find the global optima for interpolation problem.

The structure of the paper is organized as following: Section II discusses existing methods for MDS and interpolation problems; Section III introduces and explains the weighted solution for MDS; The proposed weighted and DA solution for interpolation is introduced in Section IV; In Section V, we present our experiment results on 3 types of sequence dataset and compare our proposed solutions to other existing methods; Followed by our conclusion and future work in Section VI.

II. Related work

Many MDS algorithms have been proposed in the past few decades. Newton's method is used by [11] as a solution to minimize the STRESS in (1) and SSTRESS in (2). This method used the Hessian to form a basic Newton iteration, and then iterated through it until convergence. Although the time complexity of its conversion is quadratic, both Hessian construction and inversion require cubic time complexity. Quasi-Newton [12] method is proposed to solve this problem by using an approximation of inverse Hessian at each iteration. This significantly reduced the time complexity of Newton method to sub-cubic. [13] has proposed an Multi-Grid MDS (MG-MDS) to solve the isometric embedding problems. As a parallel solution, it shows the dramatic increase in performance compared to other existing methods. Scaling by Majorizing a Complicated Object Function (SMACOF) [14] is a gradient-decent-type of algorithm which is widely used for large-scale MDS problems. However, it involves full matrix inversion before the calculation with weighting, which always has cubic time complexity. Additionally, as this method is an Expectation Maximization (EM) like problem, it is suffered from local optima problem. [15] has added a DA solution to SMACOF, so called DA-SMACOF, where it increased mapping quality and decreased the sensitivity with respect to initial configuration. Simulated Annealing and Genetic Algorithm have also been used to avoid the local optima in MDS [16] [17]. However, they suffered from long running time due to their Monte Carlo approach.

As MDS requires quadratic memory to compute, it becomes a limitation for large-scale data, e.g. millions of sequences while the computing power is limited. To address this issue, many algorithms have been developed to extend the capability of various dimension reduction algorithms by embedding new points with respect to previously configured points, or known as *out-of-sample* problem. A generalized *out-of-sample* solution has been provided by [18] that uses coordinate propagation for non-linear dimension reduction, such as MDS. [19] has proposed a solution as an *out-of-sample* extension for the algorithms based on the latent variable model. In MDS, the *out-of-sample* problem could also be considered as unfolding problem since only pairwise dissimilarities between *in-sample* sequences and *out-of-sample* sequences are observed [20]. An out-of-sample extension for the Classical Multidimensional Scaling (CMDS) has been proposed in [21]. It has applied linear discriminant analysis to the labeled objects in the representation space. In contrast to them, [7] has proposed an EM-like optimization solution, called MI-MDS to solve the problem with STRESS criteria in (26), which found embedding of approximating to the distance rather than the inner product as in CMDS. In addition to that, [6] has proposed a heuristic method, called HE-MI, to lower the time cost of MI-MDS. An Oct-Tree structure called Sample Sequence Partition Tree is used in HE-MI to partition the *in-sample* 3D space, and then interpolated the *out-of-sample* data hierarchically to avoid additional time cost. However, both of the methods suffer from local optima problem as same as in SMACOF, and could only process non-weighted data.

III. WEIGHTED SOLUTION FOR DA-SMACOF

In this section, we propose a weighted solution for DA-SMACOF, a DA and weighted solution for MI-MDS. MDS and DA will be briefly discussed first, followed by introduction of WDA-SMACOF and WDA-MI-MDS.

A. *Multidimensional Scaling*

MDS is a set of statistic techniques used in dimension reduction. It is a general term for these techniques to apply on original high dimensional data and reduce their dimensions to target dimension space while preserving the correlations, which is usually Euclidean distance calculated from the original dimension space from the dataset, between each pair of data points as much as possible. It is a non-linear optimization problem in terms of reducing the difference between the mapping of original dimension space and target dimension space. In bioinformatics data visualization, each sequence in the original dataset is considered as a point in both original and target dimension space. The dissimilarity between each pair of sequences is considered as Euclidean distance used in MDS.

Given a data set of N points in original space, a pairwise distance matrix Δ can be given from these data points ($\Delta = [\delta_{ij}]$) where δ_{ij} is the dissimilarity between point i and point j in original dimension space which follows the rules: (1) Symmetric: $\delta_{ij} = \delta_{ji}$. (2) Positivity: $\delta_{ij} > 0$. (3) Zero Diagnosal: $\delta_{ii} = 0$. Given a target dimension L, the mapping of points in target dimension can be given by an $N \times L$ matrix X, where each point is denoted as x_i from original space is represented as ith row in X.

The object function represents the proximity data for MDS to construct lower dimension space is called STRESS or SSTRESS, which are given in (1) and (2):

$$\sigma(X) = \sum_{i<j\leq N} w_{ij} \left(d_{ij}(X) - \delta_{ij}\right)^2 \tag{1}$$

$$\sigma(X) = \sum_{i<j\leq N} w_{ij} \left(d_{ij}^2(X) - \delta_{ij}^2\right)^2 \tag{2}$$

where w_{ij} denotes the possible weight from each pair of points that $w_{ij} \geq 0, W = [w_{ij}]$, d_{ij} denotes the Euclidean distance between point i and j in target dimension. Due to the non-linear property of MDS problem, an EM-like optimization method called SMACOF is proposed to minimize the STRESS value in (1). And to overcome the local optima problem mentioned previously, [15] added a computational temperature to the SMACOF function, called DA-SMACOF. It has been proved to be reliable, fast, and robust without weighting.

B. *Deterministic Annealing*

DA is an annealing process that finds global optima of an optimization process instead of local optima by adding a computational temperature to the target object function. By lowering the temperature during the annealing process, the problem space gradually reveals to the original object function. Different from Simulated Annealing, which is based on Metropolis algorithm for atomic simulations, it neither rely on the random sampling process nor random decisions based on current state. DA uses an effective energy function, which is derived through expectation and is deterministically optimized at successively reduced temperatures.

In DA-SMACOF, the STRESS function in (1) is used as object function. We denote the $\mathcal{H}_{MDS}$ as the cost function for SMACOF, and $\mathcal{H}_0$ as a simple Gaussian distribution:

$$\mathcal{H}_{MDS} = \sum_{i<j\leq N} w_{ij} \left(d_{ij}(X) - \delta_{ij}\right)^2 \tag{3}$$

$$\mathcal{H}_0 = \sum_{i<j\leq N} (x_i - u_i)^2/2 \tag{4}$$

where u_i is the average of simple Gaussian distribution of ith point in target dimension L. Also, the probability distribution $\mathcal{P}_0$ and free energy $\mathcal{F}_0$ are defined as following:

$$\mathcal{P}_0 = \exp(-\mathcal{H}_0/T + \mathcal{F}_0/T) \tag{5}$$

$$\mathcal{F}_0 = -T ln \int \exp(-\mathcal{H}_0(X)/T) dX \tag{6}$$

$$= -T ln(2\pi T)^{L/2} \tag{7}$$

where T is the computational temperature used in DA.

C. *Weighted DA-SMACOF*

The goal of DA in SMACOF is to minimize $\mathcal{F}_{mds}(P_0) = < \mathcal{H}_{mds} - \mathcal{H}_0 >_0 + \mathcal{F}_0(P_0)$ with respect to parameters u_i is independent of $< \mathcal{H}_0 >_0 + \mathcal{F}_0(\mathcal{P}_0)$, so the problem can be simplified to minimize $< \mathcal{H}_{mds} >$ if we ignore the terms independent of u_i. By differentiating (7), we can get

$$< (x_i - x_j)^2 > = (u_i - u_j)^2 + 2TL \tag{8}$$

where x_i is the ith point in the target dimension L, as same as ith line in matrix X.

Take (8) into (3), finally the $\mathcal{H}_{MDS}$ became

$$\mathcal{H}_{MDS} = \sum_{i<j\leq N} w_{ij} (\sqrt{(u_i - u_j)^2 + 2TL} - \delta_{ij})^2 \tag{9}$$

$$= \sum_{i<j\leq N} w_{ij} (|u_i - u_j| + \sqrt{2TL} - \delta_{ij})^2 \tag{10}$$

As the original cost function and target dimension configuration gradually changes when the computational temperature changes, we denote X_T as the target dimensional configuration and Δ_T as the dissimilarities of each pair of sequences under temperature T. So the updated STRESS function of DA-SMACOF becomes

$$\sigma(X_T) = \sum_{i<j\leq N} w_{ij} (d_{ij}(X_T) - \tilde{\delta}_{ij})^2 \tag{11}$$

where $\tilde{\delta}_{ij}$ is defined as

$$\tilde{\delta}_{ij} = \begin{cases} \delta_{ij} & if\ w_{ij} = 0 \\ \delta_{ij} - \sqrt{2TL} & else\ if\ \delta_{ij} > \sqrt{2TL} \\ 0 & other\ wise \end{cases} \tag{12}$$

Note that if the distance between point i and point j is missing fromΔ, then$w_{ij} = 0$. There is no difference between $\tilde{\delta}_{ij}$and δ_{ij} since both of the distances are considered missing values. This is not proposed in the original DA-SMACOF where all weights for all distances in Δ are set to 1.

By expanding (11), updated STRESS value can be defined as

$$\sigma(X_T) = \sum_{i<j\leq N} w_{ij}\, \tilde{\delta}_{ij}^2 + \sum_{i<j\leq N} w_{ij}\, d_{ij}^2(X_T) - 2 \sum_{i<j\leq N} w_{ij}\, \tilde{\delta}_{ij} d_{ij}(X_T) \tag{13}$$

$$= \eta_{\tilde{\delta}}^2 + \eta^2(X_T) - 2\rho(X_T) \tag{14}$$

Algorithm 1 WDA-SMACOF algorithm

```
Input: $\Delta$, $X$, $\varepsilon$ and $\alpha$
Generate random initial mapping $X$.
$k = 0$;
while $T_k \geqslant T_{min}$ do
  Compute $T_k$ and $\tilde{\Delta}_k$ using (12).
  $t = 0$;
  while $\sigma(X^{|t|}) - \sigma(X^{|t+1|}) > \varepsilon$
    Use CG defined from (26) to (30) to solve (23).
    $t = t + 1$;
  end while
  Cool down computational temperature $T_{k+1} = \alpha T_k$;
  $k = k + 1$
end while
$X$ =output of SMACOF based on $T_k$
return $X$
```

Equation (14) has three terms, the first term $\eta_{\tilde{\delta}}^2$ is a constant because it only depends on fixed weights and temperature, so it is a constant. Then to obtain the majorization algorithm for $\eta^2(X_T)$ and $\rho(X_T)$, they are defined as following:

$$\eta^2(X_T) = tr\ X_T' V X_T \tag{15}$$

$$\rho(X_T) = tr\ X_T' B(X_T) X_T \tag{16}$$

where V and $B(X_T)$ is defined as following:

$$v_{ij} = \begin{cases} -w_{ij} & if\ i \neq j \\ -\sum_{k\neq i} v_{ik} & else\ if\ i = j \end{cases} \tag{17}$$

$$b_{ij} = \begin{cases} -\frac{w_{ij}\tilde{\delta}_{ij}}{d_{ij}(X_T)} & if\ i \neq j, w_{ij} \neq 0, and\ d_{ij}(X_T) \neq 0 \\ -\sum_{k\neq i} b_{ik} & else\ if\ i = j \\ 0 & otherwise \end{cases} \tag{18}$$

Finally, to find the majorizing function for (11), we apply (15) and (16) to (14). By using *Cauchy-Schwarz* inequality, the majorization inequality for the STRESS function is obtained as following

$$\sigma(X_T) = \eta_{\tilde{\delta}}^2 + tr\ X_T' V X_T - 2tr\ X_T' B(X_T) X_T \tag{19}$$

$$\leq \eta_{\tilde{\delta}}^2 + tr\ X_T' V X_T - 2tr\ X_T' B(Z_T) Z_T = \tau(X_T, Z_T) \tag{20}$$

By setting the derivatives of $\tau(X_T, Z_T)$ to zero, we finally get the formula of the WDA-SMAOCF,

$$V X_T = B(Z_T) Z_T \tag{21}$$

$$X_T^u = V^\dagger B(Z_T) Z_T \tag{22}$$

where $V^\dagger$ is the pseudo-inverse of V. And Z_T is the estimated X_T from previous iteration. Equation (22) is also called Guttman transform by De Leeuw and Heiser [14]. Although $V^\dagger$ could be calculated separately from SMACOF algorithm since V is static during the iterations, the time complexity of full rank matrix inversion is always $O(N^3)$ [22][23]. Compared to the time complexity of SMACOF, which is $O(N^2)$, this is bottleneck for large-scale computation of weighted SMACOF.

Instead of using pseudo-inverse of V, we denote $V + I$ as $\dot{V}$ and if N is large, $V + I \approx V$, where I is an $N \times N$ identity matrix, so by replacing V by $\dot{V}$ in (21), we have the majorizing function of WDA-SMACOF as

$$\dot{V}X_T = B(Z_T)Z_T \tag{23}$$

Theorem 1. $\dot{V}$ is a symmetric positive definite (SPD) matrix.

Proof. Since $w_{ij} = w_{ji}$, so $v_{ij} = v_{ji}$, and $\dot{V} = \dot{V}^T$. From (17), $\dot{V}$ can be represented as

$$\dot{v}_{ij} = \begin{cases} -w_{ij} & if\ i \neq j \\ 1 + \sum_{k \neq i} w_{ik} & else\ if\ i = j \end{cases} \tag{24}$$

Because $w_{ij} \geq 0$, so $\dot{v}_{ii} > 0$. And $\dot{v}_{ii} > \sum_{k \neq i} w_{ik} = \sum_{k \neq i} |\dot{v}_{ik}|$. So according to [24], Theorem 1 is proved.

Since $\dot{V}$ is an SPD matrix, we could solve (23) instead of (22) without doing the pseudo-inverse of V. To address this issue, a well-known iterative approximation method to solve the $Ax = b$ form equation, so called Conjugate Gradient (CG) [25] could be used here. Traditionally, it is used to solve quadratic form while x and b are both vectors. In our case, $B(Z)Z$ and X are both $N \times L$ matrices. So the original CG could be directly used when $L = 1$. Nevertheless, for $L > 1$ situations, the CG method needs to be updated using following equations. In ith iteration of CG, the residual is denoted as r_i, the search direction is denoted as d_i, α_i and β_i are scalars. So r_0 and d_0 are given as

$$r_0 = d_0 = \dot{B} - \dot{V}X \tag{25}$$

where $\dot{B}$ is the produce of $B(Z) \times Z$.

Let's denote $dot(X, Y) = \sum_{1 \leq j \leq L} \sum_{1 \leq i \leq N} x_{ji} y_{ij}$ where X is $L \times N$ and Y is $N \times L$ matrix and x_{ji} is the jth row, ith column element in X and y_{ij} is the ith row, jth column element in Y. In another word, $dot(X, Y)$ is calculating the sum of dot product over rows of X and their corresponding columns in Y. So the complete equations for CG are updated to

$$\alpha_i = \frac{dot(r_i^t, r_i)}{dot(d_i^t, \dot{V}d_i)} \tag{26}$$

$$X_{i+1} = X_i + \alpha_i d_i \tag{27}$$

$$r_{i+1} = r_i - \alpha_i \dot{V} d_i \tag{28}$$

$$\beta_{i+1} = \frac{dot(r_{i+1}^t, r_{i+1})}{dot(r_i^t, r_i)} \tag{29}$$

$$d_{i+1} = r_{i+1} + \beta_{i+1} d_i \tag{30}$$

It is a recognized fact that original CG is an iterative algorithm, that X and the other parameters are updated in each iteration. And the error, which is denoted as $f(X) = \dot{B} - \dot{V}X$, is a non-increasing value until converge. So the time complexity of CG is $O(N^2)$ as the matrix multiplication in (26) and (28) are $N \times N \times L$ where $L \ll N$.

WDA-SMACOF algorithm is illustrated in Algorithm 1. The initial temperature $T_0$is critical in WDA-SMACOF that a flat initial configuration (all distance in Δ_{T_0} equals to zero) needs to be avoided. So the T_0 is calculated based on maximum value of weight times distance.

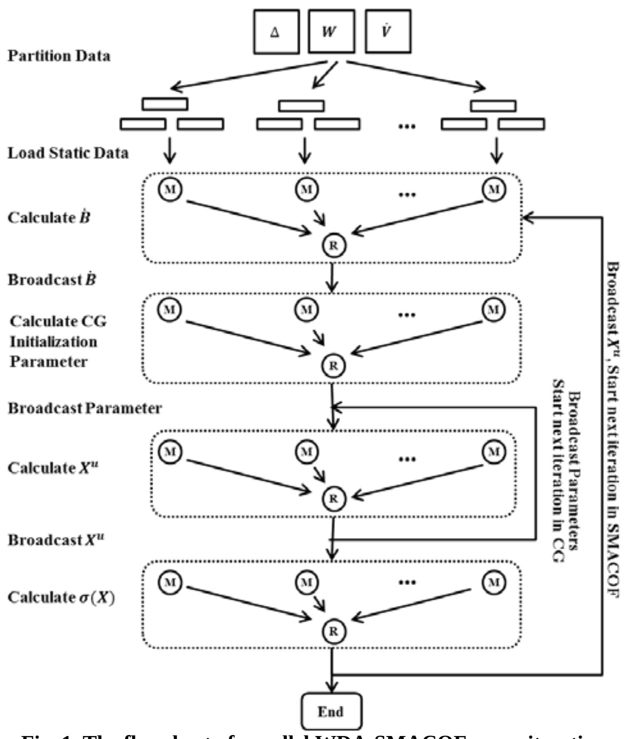
Δ
W
V̇
Partition Data
Load Static Data
Calculate Ḃ
M
R
Broadcast Ḃ
Calculate CG Initialization Parameter
Broadcast Parameter
Calculate X^u
Broadcast X^u
Calculate σ(X)
End
Broadcast X^u, Start next iteration in SMACOF
Broadcast Parameters Start next iteration in CG

Fig. 1. The flowchart of parallel WDA-SMACOF on an iterative MapReduce runtime. M denotes mapper, R denotes reducer.

The original SMACOF uses an $O(N^3)$ matrix inversion first, then do an $O(N^2)$ matrix multiplication in each iteration. WDA-SMACOF does the same $O(N^2)$ matrix multiplication, and one CG approximation in each SMACOF iteration as well. Therefore, WDA-SMACOF has a much higher scalability than original SMACOF proposed as Guttman transform.

D. Parallezation of WDA-SMACOF

As WDA-SMAOCF is an iterative optimization algorithm, we use an iterative MapReduce runtime, called Twister [26], to parallelize it for maximum performance. We also improved the overall performance of DACIDR by using a hybrid MapReduce workflow system [27]. Note that different from DA-SMACOF, the weight matrix W, and $\dot{V}$ are included during the computation, so the memory usage of WDA-SMAOCF is higher compared to DA-SMAOCF. However, since both $W$and $\dot{V}$ are $N \times N$ matrices, WDA-SMACOF still has memory (space) complexity of $O(N^2)$.

The parallelized WDA-SMACOF uses three single MapReduce computations and one nested iterative MapReduce computation in one iteration as outer loop. The nested iterative MapReduce computation has one single MapReduce computation in one iteration as inner loop. The computations in outer loop contain two matrix multiplication and one STRESS calculation. The computation in inner loop performs approximation in CG, as illustrated in Figure 1. The largest parameter broadcasted in each iteration is an $N \times L$ matrix, where L is often set to 2 or 3 for visualization purpose. So there is not much communication overhead with this design.

Algorithm 2 WDA-MI-MDS algorithm

```
Input: $D_1$, $D_2$, $X_1$, $\varepsilon$ and $\alpha$
for each $\hat{x}$ in $X_2$ do
  Compute $\hat{\Delta}$ and $\hat{X}_1$
  Compute $T_0$
  $k = 0$;
  while $T_k \geqslant T_{min}$ do
    Update $\hat{\Delta}$ to $\tilde{\Delta}$ using (12)
    Initialize random mapping for $\hat{x}$, called $\hat{x}^{|0|}$
    $t = 0$
    while $\sigma(\hat{x}^{|t|}) - \sigma(\hat{x}^{|t+1|}) > \varepsilon$
      Update $\hat{x}^{|t+1|}$ using (45)
      $t = t + 1$
    end while
    Cool down computational temperature $T_{k+1} = \alpha T_k$
    $k = k + 1$
  end while
end for
return $X_2$
```

IV. Weighted DA solution for MI-MDS

In this section, we propose a weighted deterministic annealing solution for MI-MDS. First, we briefly discuss *out-of-sample* problem and MI-MDS, and then we describe weighted DA-MI-MDS (WDA-MI-MDS) in detail.

A. *Out-of-Sample Problem and MI-MDS*

The *in-sample* and *out-of-sample* problem has been brought up in data clustering and visualization to solve the large-scale data problem. In DACIDR, MDS is used to solve the *in-sample* problem, where a relatively smaller size of data is selected to construct a low dimension configuration space. And remaining *out-of-sample* data can be interpolated to this space without the usage of extra memory.

In formal definition, suppose we have a dataset contains size of N *in-sample* data, denoted as D_1, and size of M *out-of-sample* points, denoted as D_2, where *in-sample* points were already mapped into an *L-dimension* space, and the *out-of-sample* data needs to be interpolated to an *L-dimension* space, defined as $X = \{X_1, X_2\}$, where $X_1 = \{x_1, x_2, x_3, \dots, x_N\}$ and $X_2 = \{x_{N+1}, x_{N+2}, x_{N+3}, \dots, x_{N+M}\}$. Note that only one point at a time is interpolated to the *in-sample* space. So the problem can be simplified to interpolate a point $\hat{x}$ to *L-dimension* with the distance observed to *in-sample* points. The STRESS function for $\hat{x}$ is given by

$$\sigma(X) = \textstyle\sum_{i \le N} w_{i\hat{x}} (d_{i\hat{x}}(X) - \delta_{i\hat{x}})^2 \tag{31}$$

where $d_i(\hat{x})$ is the distance from $\hat{x}$ to *in-sample* point i in target dimension, and $\delta_{i\hat{x}}$ is the original dissimilarity between $\hat{x}$ and point i. If all weights equals to 1, equation (31) is transformed to

$$\sigma(X) = \textstyle\sum_{i \le N} (d_{i\hat{x}}(X) - \delta_{i\hat{x}})^2 \tag{32}$$

MI-MDS is an iterative majorization algorithm proposed by [7] to minimize the STRESS value in (32), where all weights are assumed to be 1. It will find k nearest neighbors from *in-sample* points of a given *out-of-sample* point $\hat{x}$ at first, denoted as $P = \{p_1, p_2, p_3, \dots, p_k\}$. Then by finding a majorizing function, its minimum STRESS can be obtained analytically.

B. *Weighted DA solution for Majorizing Interpolation MDS*

MI-MDS has been proved to be efficient when deal with large-scale data. However, there are two disadvantages with this method. First, it assumed that all weights equal to one, where it couldn't deal with missing values and different weights. Secondly, this method is an EM-like optimization algorithm, which could be trapped in local optima as the EM-SMACOF.

Therefore, we propose WDA-MI-MDS to solve these issues. To solve the weighted *out-of-sample* problem, we need to find an optimization function for (31). By expanding (31), we have

$$\sigma(X) = \sum_{i \le N} w_{i\hat{x}} \delta_{i\hat{x}}^2 + \sum_{i \le N} w_{i\hat{x}}\, d_{i\hat{x}}^2(X) + 2 \sum_{i \le N} w_{i\hat{x}} \delta_{i\hat{x}} d_{i\hat{x}} \tag{33}$$

$$= \eta_\delta^2 + \eta^2(X) - 2\rho(X) \tag{34}$$

where η_δ^2 is a constant irrelevant to X. So similar to SMACOF, only $\eta^2(X)$ and $\rho(X)$ need to be considered to obtain the majorization function. $\eta^2(X)$ can be deployed to

$$\eta^2(X) = w_{1\hat{x}} \|\hat{x} - p_1\|^2 + \dots + w_{N\hat{x}} \|\hat{x} - p_N\|^2 \tag{35}$$

$$= \textstyle\sum_{i \le N} w_{i\hat{x}} \|\hat{x}\|^2 + \sum_{i \le N} w_{i\hat{x}} \|p_i\|^2 - 2\hat{x} q^t \tag{36}$$

where $q^t = (\sum_{i \le N} w_{ix} p_{i1}, \dots \sum_{i \le N} w_{ix} p_{iL})$where L is the target dimension. The *Cauchy-Schwarz* inequality can be applied on $-d_{ix}$ in $\rho(X)$ to establish the majorization function, which is given as

$$-d_{ix} = -\|\hat{x} - p_i\| \tag{37}$$

$$\le -\frac{\sum_{a \le L} (p_{ia} - \hat{x}_a)(p_{ia} - z_a)}{d_{ix}} \tag{38}$$

$$= -\frac{(p_i - \hat{x})^t (p_i - z)}{d_{iz}} \tag{39}$$

where z is a vector of length L which contains $(z_1, \dots, z_L)$, and $d_{ix} = \|p_i - z\|$. By applying (37) to $\rho(X)$, we will have

$$-\rho(X) \le -\sum_{i \le N} w_{ix} \delta_{ix} \frac{(p_i - \hat{x})^t (p_i - z)}{d_{iz}} \tag{40}$$

$$= -\hat{x}^t \sum_{i \le N} \frac{w_{ix} \delta_{ix}}{d_{iz}} (z - p_i) + C \tag{41}$$

where C is a constant irrelevant from x. After applying (36) and (41) to (34), we will have

$$\sigma(X) \le \eta_\delta^2 + \sum_{i \le N} w_{i\hat{x}} \|\hat{x}\|^2 + \sum_{i \le N} w_{i\hat{x}} \|p_i\|^2 - 2\hat{x} q^t - \hat{x}^t \sum_{i \le N} \frac{w_{i\hat{x}} \delta_{i\hat{x}}}{d_{iz}} (z - p_i) + C = \tau(\hat{x}, z) \tag{42}$$

As both η_δ^2 and C are constants, equation (42) is a majorization function of the STRESS that is quadratic in X.

The minimum of this function can be obtained by setting the derivatives of $\tau(\hat{x}, z)$ to zero, that is

$$2\sum_{i\leq N} w_{i\hat{x}}\,\hat{x} - 2q^t - 2\sum_{i\leq N}\frac{w_{i\hat{x}}\delta_{i\hat{x}}}{d_{iz}}(z-p_i) = 0 \tag{43}$$

$$\hat{x}^u = \frac{q^t + \sum_{i\leq N}\frac{w_{i\hat{x}}\delta_{i\hat{x}}}{d_{iz}}(z-p_i)}{\sum_{i\leq N} w_{i\hat{x}}} \tag{44}$$

where z is the previous estimated $\hat{x}$. Although this algorithm so far can guarantee to generate a series of non-increasing STRESS value for from original distances with various weights, it still could be trapped into local optima. Therefore, to add a deterministic annealing solution into that, we apply (12) to (44), and finally we have the iterative majorization equation for WDA-MI-MDS in (45), and the algorithm is illustrated in algorithm 2.

$$\hat{x}^u_t = \frac{q^t + \sum_{i\leq N}\frac{w_{i\hat{x}}\tilde{\delta}_{i\hat{x}_t}}{d_{iz}}(z-p_i)}{\sum_{i\leq N} w_{i\hat{x}}} \tag{45}$$

where $\tilde{\delta}_{ix_t}$ can be obtained using (12).

C. Parallelization of WDA-MI-MDS

Different from WDA-SMACOF, the *out-of-sample* point's dimension reduction result only depends on the *in-sample* points. So D_1 and X_1 are copied and loaded into memory on every mapper. Since every *out-of-sample* point is independent from any other *out-of-sample* points, WDA-MI-MDS can be pleasingly paralleled. Therefore D_2 is partitioned and distributed across the mappers. And the result of each mapper could be simply merged into X_2, as illustrated in Figure 2.

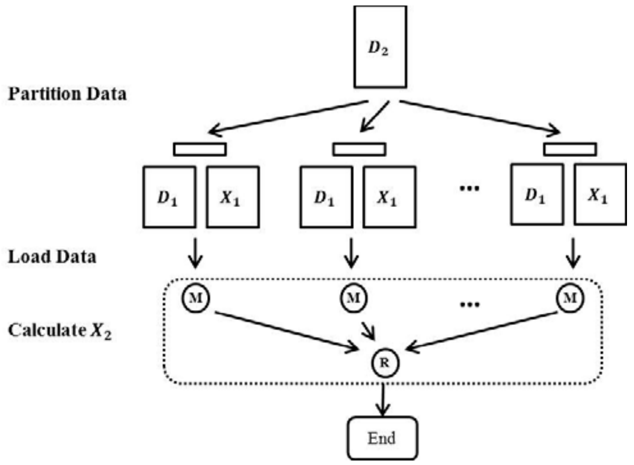

Fig. 2. The flowchart of parallel WDA-MI-MDS on MapReduce runtime. M denotes mapper, R denotes reducer.

V. Experiments

The experiments were carried out on FutureGrid XRay Cluster, which has 168 AMD Opteron 2378 CPUs and 1324 cores. We tested the accuracy of the results based on normalized STRESS value, which can be calculated by

$$\bar{\sigma}(X) = \sum_{i<j\leq N} w_{ij}\frac{(d_{ij}(X)-\delta_{ij})^2}{\delta_{ij}^2} \tag{46}$$

where δ is given by PID distance calculated from pairwise sequence alignment. Equation (46) is least squares sum of difference between the mapped distance after dimension reduction and original distance and naturally lower normalized STRESS means better performance.

In our experiments, we denote Full MDS as the algorithms performed on *in-sample* data, which runs SMACOF based on given pairwise dissimilarities; and Interpolation as the algorithms experimented on *out-of-sample* data. These *out-of-sample* data were interpolated to Full MDS result of *in-sample* data using MI-MDS within the same dataset. Full MDS includes weighted Deterministic Annealing (WDA), non-weighted Deterministic Annealing (NDA), weighted Expectation Maximization (WEM), and non-weighted Expectation Maximization (NEM) of SMACOF. Among them, WEM- and NEM-SMACOF are proposed in [6], NDA-SMACOF was proposed in [15]. Interpolation includes WDA, WEM, NDA, and NEM of MI-MDS. NEM-MI-MDS was proposed in [14], WEM-MI-MDS is implemented without DA function, and NDA-MI-MDS is implemented without Weight function. Additionally, equation (12) shows that all the EM cases could be considered as a special case of DA algorithms that initial temperatures were set to 0.

A. Nomalized Stress Comparison

In this experiment, we used three different bioinformatics dataset, include Metagenomics DNA, hmp16SrRNA and COG proteins. We compared the normalized STRESS values from Full MDS and Interpolation of all algorithms. The threshold ε is set to 10^{-6}, and the experiments were carried out by 20 runs for each algorithm. The results were based on the average of these runs. The error bars in Figure 3, Figure 4, Figure 5 and Figure10 were the maximum and minimum value in the runs.

1) Metagenomics DNA: This dataset includes 4640 unique DNA sequences. 2000 of these sequences were selected as *in-sample* data, and rest 2640 sequences were considered as *out-of-sample* data. As this dataset is relatively small, we tested the sequential version of both Full MDS and Interpolation. This sequences were aligned using local alignment algorithm to calculate the original dissimilarity. And during that calculation, 10.775% of the original distances values were found as missing because of the low alignment quality. From the result shown in Figure 3, we observed that both of the weighted solutions outperforms the non-weighted solution. DA solutions showed much less divergence compared to EM solutions. The average normalized STRESS value for Full MDS was 0.0439, which outperforms non-weighted cases by 23%.

2) hmp16SrRNA: The original hmp16SrRNA dataset has 1.1 million sequences, which was clustered and visualized by DACIDR in our previous research [5]. In this experiments, we selected 10k of it as *in-sample* data and 40k as *out-of sample* data. Due to the larger size, it can not be done on a single core, so we used the parallel version of Full MDS and Interpolation to run the experiments on 80 cores. The distance was calculated using local alignments and 9.98% of distances were

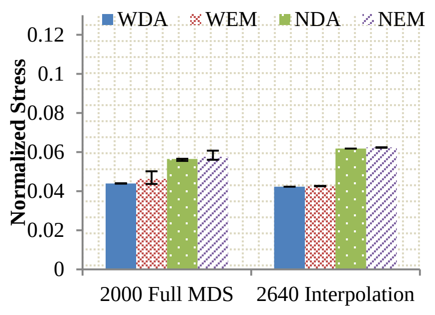

Fig. 3. The normalized STRESS comparison of Metagenomics DNA mapped into 3D. 2000 sequences were selected as in-sample data to run full MDS, and 2640 sequences were out-of-sample data runs interpolation.

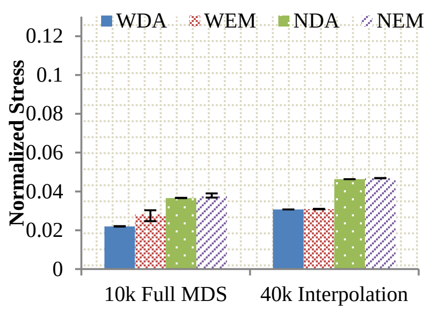

Fig. 4. The normalized STRESS comparison of hmp16SrRNA data mapped into 3D. 10k sequences were selected as in-sample data to run full MDS, and 40k sequences were out-of-sample data runs interpolation.

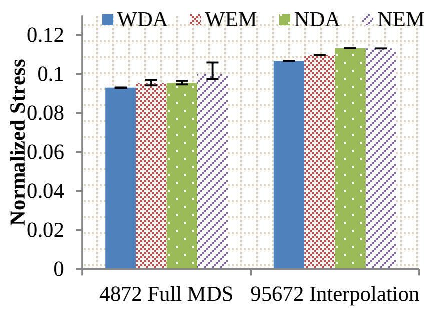

Fig. 5. The normalized STRESS comparison of COG Protein data mapped into 3D. 4872 consensus sequences were in-sample data runs full MDS, and 95672 COG sequences were out-of-sample data runs interpolation

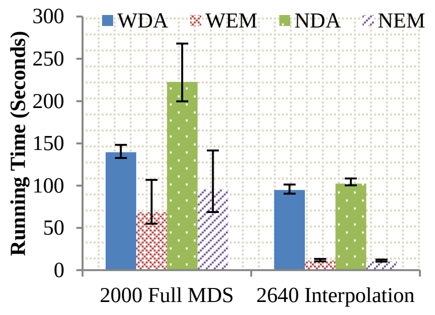

Fig. 6. The sequential running time for Metagenomics DNA mapped into 3D. The threshold is set to 10^{-6}. 2000 sequences were in-sample and 2640 were out-of-sample data.

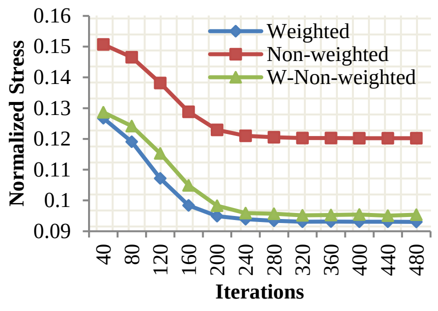

Fig. 7. The normalized STRESS comparison of Full MDS running on 4872 COG protein data at increasing iterations. Larger iteration number means longer time to process.

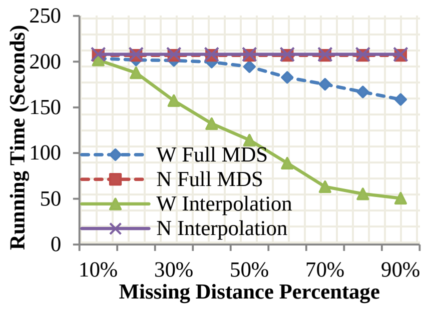

Fig. 8. The running time for parallel Full MDS on 10k and Interpolation on 40k of hmp16SrRNA data. W is short for weighted, and N is short for non-weighted.

randomly missing and set to an arbitrary number. The normalized STRESS were shown in Figure 4. In this case, the weighted solutions has a normalized STRESS value lower than non-weighted solutions by 40%.

3) COG Protein: Differently from DNA and RNA data, the Protein data doesn't have nicely clustered structure after dimension reduction, and its distance calculation was based on global alignment other than local alignment. In our experiments, we used 4872 consensus sequences to run full MDS, and interpolated rest 95672 sequences to these consensus sequences. Among these distances from Full MDS and Interpolation, 10% of them were randomly chosen to be missing distances. The runs for 4872 *in-sample* sequences were carried out on a single core, while the Interpolation for 95672 *out-of-sample* sequences used 40 cores. The results for COG Protein data were shown in Figure 5. Because of its unique configuration in 3D space, the non-weighted and weighted cases show insignificant difference that WDA performs only 7.2% better than non-weighted cases.

In these experiments, different dataset shows different features after dimension reduction. Figure 11, Figure 12, and Figure 13 are the clustering and visualization results for these three dataset shown in software called PlotViz [28]. It is clear that the Metagenomics DNA data has well-defined boundaries between clusters; the sequences in hmp16SrRNA dataset are not as clearly separated but we could still observe some clusters; COG data points were evenly distributed in the 3D space, and the different colors are indication of existence of clusters identified by [2]. Although these three dataset had diverted visualization results, WDA solution always shows lowest normalized STRESS value and smallest divergence in all experiments.

B. *Comparison of Computational Complexities*

In these results, we assume that distances δ_{ij} are calculated beforehand, and the time of CG is compared separately with full matrix inversion in subsection C. So in this section, only performance differences of computing majorizing functions in different algorithms are shown. Therefore, for all of the algorithms, the time costs reflect the number of SMACOF iterations.

1) Fixed threshold runs: For the runs in Section A where the ending condition for the algorithms wass threshold, the iteration number could be various due to the configuration/feature space of different dataset. As shown in Figure 6, for 4640 DNA data, DA solutions took longer to process because it converged multiple times as the temperature cools down. weighted solutions had less iterations because the configuration space with weight enabled faster convergence, so the total time cost of weighted solutions were smaller than non-weighted solutions. However, this effect was not permanent on different dataset. When SMACOF ran on COG data, non-weighted solutions had less iterations to converge than weighted solutions as shown in Figure 7. This feature shows that if the points in the target dimension space are almost evenly spreaded out, the iterations converge

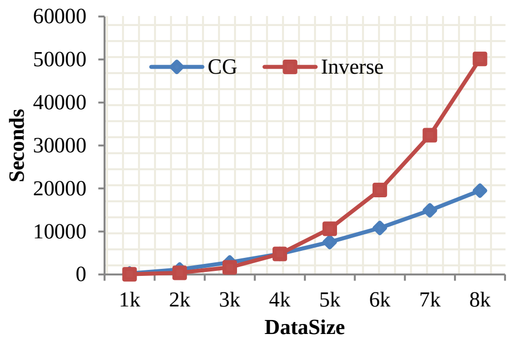

Fig. 9. The running time of CG compared to matrix inverse in SMACOF. Total iteration number of SMACOF is 480, and data is selected from hmp16SrRNA. CG has an average of 30 iterations.

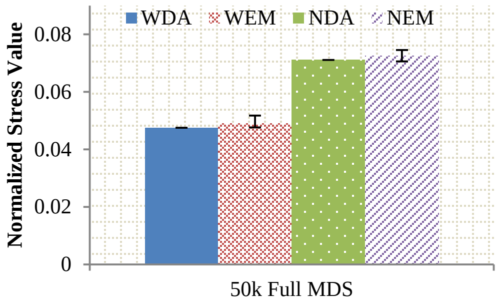

Fig. 10. The normalized STRESS comparison of hmp16SrRNA data mapped into 3D. 50k sequences were selected as in-sample data to run full MDS with Conjugate Gradient method.

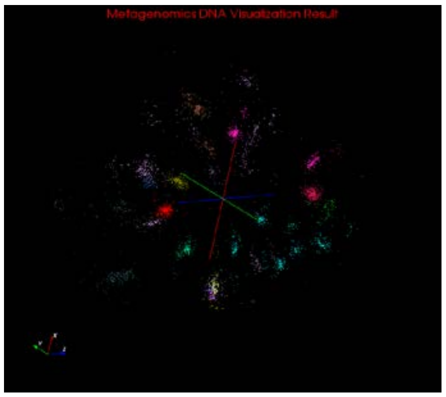

Fig. 11. Clustering and visualization result of Metagenomics DNA dataset with 15 clusters.

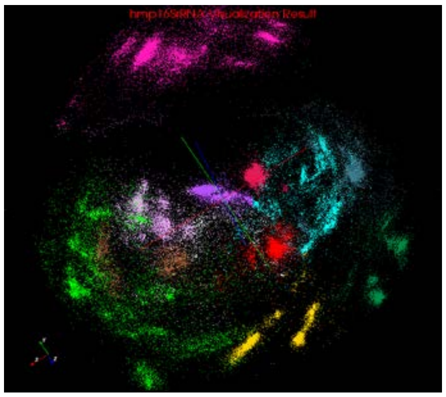

Fig. 12. Clustering and visualization result of hmp16SrRNA dataset with 12 clusters.

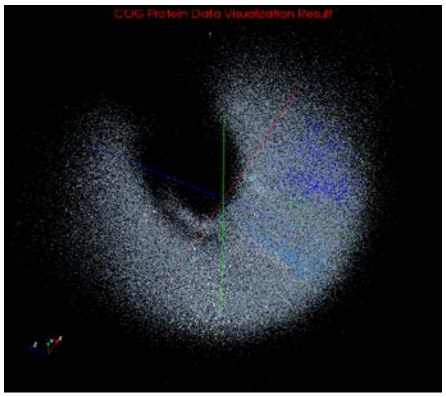

Fig. 13. Visualization result of COG protein dataset, with 11 clusters identified.

quicker. Figure 7 shows the normalized STRESS value for Full MDS of COG data as the iteration number increases. Both algorithms were set to run 480 iterations. It shows that for weighted case, the normalized STRESS kept decreasing and it finally converges after 480 iterations with a threshold of 10^{-6}. And for non-weighted case, the algorithm treats the input data as if there is no value missing. So when we calculated STRESS with non-weighted solution, it was always much higher than weighted case. It converges at about 280 iterations, but its weighted STRESS (W-non-weighted) value was still higher than WEM cases at that point.

2) Fixed iterations: If the iteration number of each run was fixed, we could simply compare the efficiency of different algorithms. Figure 8 shows how the time cost varied for weighted and non-weighted solutions of Full MDS and Interpolation when percentage of missing distance values from input increases. Full MDS ran a fixed number of iterations at 480 and Interpolation runs 50 iterations for every *out-of-sample* point. Because in non-weighted solutions, all weights were uniformly set to 1, there was no time difference for non-weighted solutions when percentage of missing distance values increased. However, for weighted solutions, if an input distance was missing, the correspond weight equaled zero. According to (18), part of calculations in Full MDS were eliminated, and as in (45), a large percentage of calculations in Interpolation weren't needed because the product of zero was still zero. The results showed that non-weighted Full MDS took an average of 206 seconds and non-weighted Interpolation took 207 seconds to finish for all cases. And weighted Full MDS only decreases 23% compared to non-weighted solution, even in case where 90% values of input distance matrix were missing. But for Interpolation, as main part of the computation were spared due to the missing values, the time cost decreases almost linearly when the percentage of missing distances increases. So it is clear that weighted solution has a higher efficiency on Full MDS and Interpolation than non-weighted solutions with fixed iterations.

In conclusion, the weighted solution is not always faster than non-weighted solution when the threshold is fixed. This happens because the algorithms may converge in different number of iterations. But if the number of iterations is fixed, the weighted problem solution has a lower time cost compared to the non-weighted case. Within a given time, the weighted solution can finish more iterations than the non-weighted solution.

C. Scalability Comparison

In this section, we did a scale up test on a single core with matrix inversion and CG to show their different time complexity. A large-scale experiment using 50k hmp16SrRNA data with Full MDS was carried out on 600 cores, where 20% of original distances are missing. Some preliminary analysis of using CG instead of matrix inversion were done. We found 30 iterations within CG sufficed for up to 50K points, so 30 CG iterations per SMACOF step was used in these experiments.

Figure 9 illustrates the difference between matrix inversion and SMACOF iteration time cost when data size goes up on a single machine. The original SMACOF performed better when data size was small, since matrix inversion ran only once before SMACOF iteration started. Additionally, we ran 480 iterations for SMACOF, and CG is processed in every SMACOF iteration, so it has a higher time cost when there were less than 4k sequences. But when data size increased to 8k, matrix inversion had significantly higher time cost than to CG. This suggests CG and its extensions gives an effectively $O(N^2)$ approach when $N \gg number\ of\ CG\ iteration$, while the time complexity of matrix inverse was always *O(N*^{3}*)*.

Figure 10 shows the result of Full MDS on 50k *in-sample* hmp16SrRNA data using CG method where CG only needed 4000 seconds in average to finish one run. The results shows that even at large scale, WDA-SMACOF still performed the best compared to other three methods.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed WDA-SMACOF and WDA-MI-MDS, as two algorithms for full MDS and interpolation problems with DA techniques and weighted problems. Our results showed that the WDA solution always performs best for weighted data. If the iteration number is fixed, weighted solution always takes less time to compute compared to non-weighted solutions. Additionally, we effectively reduced the time complexity of SMACOF from *O(N*^{3}*)* to *O(N*^{2}*)* by using Conjugate Gradient instead of full Matrix Inversion and showing that a few iterations were sufficient. Future work will include larger scale test, adding weight function to HE-MI [6].

ACKNOWLEDGMENT

This material is based upon work supported in part by the National Science Foundation under FutureGrid Grant No. 0910812. Our thanks to Mina Rho and Haixu Tang from Center for Genomics and Bioinformatics for providing the DNA and RNA data, and Larissa Stanberry from Seattle Children's Research Institute for providing the protein data.

REFERENCES

[1] A. Hughes, Y. Ruan, S. Ekanayake, S.-H. Bae, Q. Dong, et al. (2012). "Interpolative multidimensional scaling techniques for the identification of clusters in very large sequence sets." *BMC Bioinformatics* 13(Suppl 2): S9.

[2] L. Stanberry, R. Higdon, W. Haynes, N. Kolker, W. Broomall, et al. "Visualizing the Protein Sequence Universe." *Proceedings of ECMLS'12 of ACM HPDC 2012*, Delft, Netherlands, ACM, 2012.

[3] Ian T. Jolliffe. *Principal component analysis.* Vol. 487. New York: Springer-Verlag, 1986.

[4] Christopher M. Bishop, M. Svensén, and C. KI Williams. "GTM: The generative topographic mapping." *Neural computation* 10, no. 1 (1998): 215-234.

[5] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, et al. "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation." *Proceedings of the National Academy of Sciences* 96, no. 6 (1999): 2907-2912.

[6] Y. Ruan, S. Ekanayake, M. Rho, H. Tang, S.-H. Bae, et al. "DACIDR: deterministic annealed clustering with interpolative dimension reduction using a large collection of 16S rRNA sequences." In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pp. 329-336. ACM, 2012.

[7] S.-H. Bae, J. Y. Choi, J. Qiu, and G. C. Fox. "Dimension reduction and visualization of large high-dimensional data via interpolation." In*Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, pp. 203-214. ACM, 2010.

[8] N. L. Dempster, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B*, pp. 1–38, 1977.

[9] K. Rose, E. Gurewitz, and G. C. Fox, "A deterministic annealing approach to clustering," *Pattern Recognition Letters*, vol. 11, no. 9, pp. 589–594, 1990.

[10] H. Klock, and J. M. Buhmann. "Multidimensional scaling by deterministic annealing." In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 245-260. Springer Berlin Heidelberg, 1997.

[11] Kearsley A, Tapia R, Trosset M. "The solution of the metric STRESS and SSTRESS problems in multidimensional scaling using Newton's method." *Computational Statistics*, 13(3):369–396, 1998.

[12] Kelley CT. "Iterative Methods for Optimization. Frontiers in Applied Mathematics." *SIAM*: Philadelphia, 1999.

[13] M. M. Bronstein, A. M. Bronstein, R. Kimmel, and I. Yavneh. "Multigrid multidimensional scaling." *Numerical linear algebra with applications*13, no. 2-3 (2006): 149-171.

[14] I. Borg and P. J. Groenen, *Modern Multidimensional Scaling: Theory and Applications*. New York, NY, U.S.A.: Springer, 2005.

[15] Bae, Seung-Hee, Judy Qiu, and Geoffrey C. Fox. "Multidimensional Scaling by Deterministic Annealing with Iterative Majorization algorithm." In *e-Science (e-Science), 2010 IEEE Sixth International Conference on*, pp. 222-229. IEEE, 2010.

[16] M. Brusco, "A simulated annealing heuristic for unidimensional and multidimensional (city-block) scaling of symmetric proximity matrices," *Journal of Classification*, vol. 18, no. 1, pp. 3–33, 2001.

[17] R. Mathar and A. ˇZilinskas, "On global optimization in two-dimensional scaling," *Acta Applicandae Mathematicae: An International Survey Journal on Applying Mathematics and Mathematical Applications*, vol. 33, no. 1, pp. 109–118, 1993.

[18] S. Xiang, F. Nie, Y. Song, C. Zhang, and C. Zhang. Embedding new data points for manifold learning via coordinate propagation. *Knowledge and Information Systems*, 19(2):159–184, 2009.

[19] M. Carreira-Perpin´an and Z. Lu. The laplacian eigenmaps latent variable model. In *Proc. of the 11*th *Int. Workshop on Artificial Intelligence and Statistics (AISTATS 2007)*. Citeseer, 2007.

[20] C. H. Coombs. "A theory of data". (1950) New York: Wiley.

[21] M. W. Trosset and C. E. Priebe. "The out-of-sample problem for classical multidimensional scaling." *Computational Statistics and Data Analysis*, 52(10):4635-4642, 2008.

[22] P. F. Dubois, A. Greenbaum, and Garry H. Rodrigue. "Approximating the inverse of a matrix for use in iterative algorithms on vector processors."*Computing* 22, no. 3 (1979): 257-268.

[23] Peter D. Robinson, and Andrew J. Wathen. "Variational bounds on the entries of the inverse of a matrix." *IMA journal of numerical analysis* 12, no. 4 (1992): 463-486.

[24] Curtis F. Gerald, and Patrick O. Wheatley. *Numerical analysis*. Addison-Wesley, 2003.

[25] Van der Vorst, Henk A. "An iterative solution method for solving *f(A)x= b*, using Krylov subspace information obtained for the symmetric positive definite matrix *A*." *Journal of Computational and Applied Mathematics* 18, no. 2 (1987): 249-263.

[26] J.Ekanayake, H. Li, B. Zhang, T. Gunarathne, et al. "Twister: A Runtime for iterative MapReduce." *Proceedings of MapReduce'10 of ACM HPDC 2010*, pp. 810-818. ACM, 2010.

[27] Y. Ruan, Z. Guo, Y. Zhou, J. Qiu, and G.C. Fox. "HyMR: a Hybrid MapReduce Workflow System." *Proceedings of ECMLS'12 of ACM HPDC 2012*, Delft, Netherlands, ACM, 2012.

[28] PlotViz - A tool for visualizing large and high-dimensional data. http://salsahpc.indiana.edu/pviz3/