Accelerated Learning of Generalized Sammon Mappings

Yinjie Huang, Michael Georgiopoulos and Georgios C. Anagnostopoulos

Abstract—The Sammon Mapping (SM) has established itself as a valuable tool in dimensionality reduction, manifold learning, exploratory data analysis and, particularly, in data visualization. The SM is capable of projecting high-dimensional data into a low-dimensional space, so that they can be visualized and interpreted. This is accomplished by representing inter-sample dissimilarities in the original space by Euclidean inter-sample distances in the projection space. Recently, Kernel Sammon Mapping (KSM) has been shown to subsume the SM and a few other related extensions to SM. Both of the aforementioned models feature a set of linear weights that are estimated via Iterative Majorization (IM). While IM is significantly faster than other standard gradient-based methods, tackling data sets of larger than moderate sizes becomes a challenging learning task, as IM’s convergence significantly slows down with increasing data set cardinality. In this paper we derive two improved training algorithms based on Successive Over-Relaxation (SOR) and Parallel Tangents (PARTAN) acceleration, that, while still being first-order methods, exhibit faster convergence than IM. Both algorithms are relatively easy to understand, straightforward to implement and, performance-wise, are as robust as IM. We also present comparative results that illustrate their computational advantages on a set of benchmark problems.

I. INTRODUCTION

The Sammon Mapping (SM) is a multi-dimensional scaling technique that was introduced in [1]. Using the available data, SM learns an implicit non-linear projection from the data’s original high-dimensional space to, typically, a 2- or 3-dimensional projection space. The location of the SM images of the data are determined, so that the inter-point dissimilarities in the high-dimensional are represented as Euclidean distances in the projection space as faithfully as possible. In the case, where the aforementioned dissimilarities are also Euclidean distances, then the SM learns an approximate isometry from one space to the other. However, it is the fact that it can use almost arbitrarily defined dissimilarities that made SM a very useful and broadly-applicable method in dimensionality reduction, manifold learning, exploratory data analysis and, in particular, data visualization.

The SM has found many applications. For example, it was used in the context of chromosome classification in [2]. It was also applied for visualization of reconstructed phase space trajectories of chaotic systems in [3]. Moreover, it was used for visualizing multi-listener room response equalization in [4], [5]. It was also used in electricity customer classification [6]. Besides, SM was also adopted in city models mapping [7], visualization of web usage patterns [8], classification of protein profiles [9] and visualization of transitions of hepatitis [10]. On balance, as long as the inter-point distances in the projected space lead to a meaningful interpretation, the SM can be an invaluable exploratory/visualization tool.

The original SM lacks the ability to interpolate and extrapolate data, that have not been used for its design. This is because its adjustable parameters are directly the data’s projections. Methods were developed to overcome this drawback by assuming that the projections are generated by specific parameterized models. Notable efforts along this path are SAMMAN [11] and the work of deRidder and Duin [12], both of which utilize a Multi-layer Perceptron (MLP) to learn the embedding map. Additionally, generating the projections via a Radial Basis Function (RBF) Neural Network was explored in [13]. More recently, [14] introduced the Kernel Sammon Mapping (KSM), which employs a linear combination of kernel bases to implement the embedding and subsumes the SM and the previously introduced MLP- and RBF-based approaches.

A common element for most of these aforementioned methods to learn the embedding function is a set of linear weights that finally produce the data’s images in the projection space. An efficient algorithm based on Iterative Majorization (IM) for estimating these weights has been devised by [15] for the SM and has been extended for what we will refer to in Section II as the Generalized Sammon Mapping (GSM) by [13] and [14]. While IM is significantly faster than gradient descent-based methods, IM is still a linearly convergent method that slows down as the number of weights increases. Note that the number of weights is typically proportional to the number of samples to be projected. This characteristic limits the application of SM and related methods to quite small sample sizes.

The aim of this paper is to explore acceleration methods to the general IM scheme, which will render the SM projection of larger data sets more practical. This can be accomplished by regarding IM as an iterative, differentiable map and applying acceleration procedures specifically designed for fixed-point maps. In this work we explore acceleration of IM based on Successive Over-Relaxation (SOR) and the Parallel Tangents (PARTAN). In specific, we adapt these methods to the original IM scheme and derive two algorithms, namely SOR Accelerated IM (SOR-IM) and PARTAN Accelerated IM (PARTAN-IM), that are relatively easy to understand and to implement. We tested these acceleration schemes on benchmark problems and provide experimental results that
show, indeed, that they can achieve significant speedup over the original IM algorithm and that, performance-wise, are equally robust.

The rest of the paper is organized as follows. In Section II we provide some important background regarding the SM, KSM and GSM and describe the IM fixed-point map. In Section III we describe the SOR and PARTAN acceleration methods and show how they can be applied to the IM procedure. Then, in Section IV we provide comparative results for the original IM algorithm and its two accelerated variants on a collection of benchmark problems. Finally, in Section V we end this paper with a summary of our conclusions.

II. THE GENERALIZED SAMMON MAPPING

Let us assume the availability of \( N \) samples \( \{x_n \in \mathbb{F}\}_{n=1}^{N} \), where \( \mathbb{F} \) is an arbitrary feature space. Let \( \delta_{ij} \) denote the dissimilarity between the \( i^{th} \) and \( j^{th} \) samples. It is assumed that these dissimilarities are symmetric in their indices and that \( \delta_{ii} = 0 \) for \( i = 1, \ldots, N \). Note, that any metric on \( \mathbb{F} \) can be used as a dissimilarity measure.

Given the aforementioned data set, both the original SM and its extension, the KSM ([14]), produce a configuration of \( N \) points \( \{y_n \in \mathbb{F}\}_{n=1}^{N} \), where \( P \ll \dim \mathbb{F} \) (typically \( P = 2, 3 \)). Each original point \( x_n \) is thought to be corresponding to its image \( y_n \). The KSM’s goal is to position these projections in such a manner, so that \( d_{ij} \approx \|y_i - y_j\|_2 \) reflects the dissimilarity \( \delta_{ij} \) as faithfully as possible. The images of the original points are generated as follows

\[
y = W^T k(x)
\]

where \( W \in \mathbb{R}^{H \times P} \) is an oblique projection matrix that maps the vector \( k \) onto the low-dimensional space \( \mathbb{F}^P \). Note that, typically, \( H \leq N \). Also, \( k \in \mathbb{F}^H \) is, in general, a non-linear mapping that may or may not be parameterized. The weights \( W \) are estimated so that the stress function, shown below, is minimized.

\[
\sigma(W) = \sum_{1 \leq i < j \leq N} u_{ij} (d_{ij} - \delta_{ij})^2
\]

where \( U \in \mathbb{R}^{N \times N} \) is a hollow symmetric, real-valued matrix with non-negative entries (i.e. \( u_{ij} = u_{ji} \geq 0 \), \( u_{ii} = 0 \) for \( i, j = 1, \ldots, N \)), that determines the importance of individual discrepancies between dissimilarities and distances in the projection space. Typically, \( U \) is an all-ones matrix, save its diagonal, which is assumed to contain entries equal to zero.

We will refer to models that perform the projection according to (1) in order to minimize the stress function in (2) as Generalized Sammon Mappings (GSMs). The SM utilizes a mapping \( k \) with \( H = N \) that is defined as

\[
k_h(x) = [x = x_h] \quad h = 1, \ldots, H
\]

where \([\cdot]\) denotes the Iversonian bracket; it equals 1, if its enclosed predicate is true, otherwise it equals 0. Because of (3), the SM is unable to produce projections for samples that do not belong to its training set and, thus, is incapable of interpolation and/or extrapolation. On the other hand, the KSM uses a mapping defined as

\[
k_h(x) = k(x, c_h | \psi) \quad h = 1, \ldots, H
\]

where \( k : \mathbb{F} \times \mathbb{F} \rightarrow \mathbb{R} \) is a Mercer (inner-product) kernel (e.g. see [16]). In this capacity, the KSM employs an implicit, bounded \( L_2 \)-norm mapping \( \phi_{x} : \mathbb{F} \rightarrow \mathbb{H} \) to a (possibly, infinite-dimensional) Hilbert space \( \mathbb{H} \), such that \( k(x, c) = \langle \phi_x, \phi_c \rangle_\mathbb{H} \). In this capacity, the KSM employs an implicit, bounded \( L_2 \)-norm mapping \( \phi_{x} : \mathbb{F} \rightarrow \mathbb{H} \) to a (possibly, infinite-dimensional) Hilbert space \( \mathbb{H} \), such that \( k(x, c) = \langle \phi_x, \phi_c \rangle_\mathbb{H} \). We assume that the kernels are parameterized by their second arguments via the vectors \( c_h \in \mathbb{F} \), which we will be referring to as prototype vectors. These vectors can be treated as model parameters or can be appropriately chosen from the training set. Optionally, all kernel functions may have a common scalar parameter \( \psi \in \mathbb{R}^+ \). The role of these kernels is to measure the similarity between a test sample and the prototype vectors.

For the KSM, the use of appropriate kernels accommodates a variety of data, including data that have categorical or mixed-type attributes, and allow for interpolation/extrapolation in a natural manner. Furthermore, many Mercer kernels are also RBFs. This opens up the possibility of using kernels that solely depend on the dissimilarities between test and prototype patterns, provided that they can be somehow computed.

Nevertheless, in the sequel we’ll assume that \( k \) is a fixed mapping that only features \( x \) as its free argument. As a matter of fact, in our experimental setup in Section IV, when we use the KSM we pick prototype vectors by sub-sampling the training set and fix \( \psi \) to a convenient value. Instead, we’ll focus on the GSM, whose only model parameter is the weight matrix \( W \), regardless of how \( k \) is being produced. This way the results we report in this paper apply to all possible SM-based methods and allows us to isolate the process and effects of efficiently estimating \( W \), as it will be shown shortly.

For the GSM, estimation of \( W \) can be accomplished via the following fixed-point iteration scheme:

\[
W_{t+1} = M(W_t) = A^T B(W_t) W_t
\]

\( A^T \) denotes the Moore-Penrose (pseudo)inverse of \( A \). The auxiliary matrices \( A \) and \( B \) are defined as follows:

\[
A \triangleq \sum_{1 \leq i < j \leq N} u_{ij} \Delta k_{ij} \Delta k_{ij}^T
\]

\[
B(W) \triangleq \sum_{1 \leq i < j \leq N} u_{ij} \delta_{ij} d_{ij}^T(W) \Delta k_{ij} \Delta k_{ij}^T
\]

In (6) and (7), we define \( \Delta k_{ij} \triangleq k(x_i) - k(x_j) \). Also, for a real-valued scalar \( d \) we define \( d^T \) as \( \frac{1}{d^2} \), if \( d \neq 0 \), and as 0, if \( d = 0 \). We’ll refer to (5) as the IM Iteration. It is derived in [13] via majorization of the stress function depicted in (2). Application of (5) for the SM produces the SMACOFF
Algorithm 1 Iterative Majorization (IM)

Input: $W_0 \neq O, \ t_{max} \geq 1, \ \tau > 0$
Output: $W_{final}$

1: for $t = 0$ to $t_{max}$ do
2: // Compute gradient matrix
3: $G_t \leftarrow [A - B(W_t)]W_t$
4: // Check for convergence
5: if $\|\text{vec}(G_t)\|_\infty \leq \tau$ then
6: $W_{final} \leftarrow W_t$
7: break
8: end if
9: // Compute IM’s search matrix
10: $D_{IM} \leftarrow -A^tG_t$
11: // Update weights
12: $W_{t+1} \leftarrow W_t + D_{IM}$
13: end for
14: $W_{final} \leftarrow W_{t_{max}}$
15: return $W_{final}$

algorithm presented in [15]. Furthermore, the KSM uses the same rule for updating its weight matrix.

A practical implementation of the estimation process involving IM is provided in Algorithm 1. vec($G$) denotes the vector obtained by orderly concatenating the columns of $G$ into a single-column vector. Note that $A^t$ needs to be computed only once before commencing the iterations. Very often, especially in the case when KSM is used as the model to generate the projections, $A$ is full-rank, and, therefore, the Moore-Penrose inverse simplifies to the ordinary matrix inverse. Moreover, IM always produces descent directions. This is because the gradient matrix $G = \frac{\partial}{\partial W}$ of the stress function $\sigma$ of (2) is given as

$$ G = [A - B(W)]W $$

and it can be shown that the IM search direction matrix $D_{IM}$ can be expressed as

$$ D_{IM} = M(W) - W = -A^tG $$

Since $A$ can be shown to be positive semi-definite, (9) illustrates that $D_{IM}$ corresponds to a descent direction in the weight-space. Another conclusion that can be drawn from (9) is the fact that a fixed point of $M$ (i.e., a $W^*$, such that $M(W^*) = W^*$) is also a stationary point of $\sigma$.

Overall, IM’s popularity stems from the fact that it encompasses several desirable characteristics. Due to its very nature (by design), IM is guaranteed to monotonically converge to a local minimum of the stress function, unless started at a stationary point of $\sigma$; curiously enough, such a point is $W^* = O$. Furthermore, there is no specific need to control its step length via a line search method. Hence, it is very straightforward to implement, as one can witness by inspecting Algorithm 1. Finally, it has proven to converge much faster than other algorithms that utilize the gradient of the stress function, such as gradient descent, (non-linear) conjugate gradient and other related methods.

III. ACCELERATION METHODS

Despite its speed advantages, IM can slow down significantly, when the number of weights (i.e., the product $HP$) increases. This typically occurs, when the training set size $N$ increases and, at the same time, high representation fidelity (i.e., low $\sigma$ values) are desired, in which case $H$ needs to be a significant fraction of $N$. This fact motivates us to seek avenues for devising iterative schemes, that are based on IM, but whose convergence speed scales more favorably, as the number of projection weights increases. Since the IM is a fixed-point map, a natural choice is to investigate acceleration methods that are specifically designed for such maps.

An obvious approach is to move further, thus, extrapolating along the IM search direction in the hope that lower values of the stress function are encountered and, long term, the total number of iterations necessary to reach the vicinity of a local minimum are reduced. At the same time, the computational complexity of the new scheme needs to be controlled, so iterations still remain relatively inexpensive. Such a scheme is offered by a non-linear version of the SOR method, which amounts to employing the following weight update rule

$$ W_{t+1} = W_t + \alpha D_{IM} $$

with $\alpha \geq 1$ as large as possible (notice that $\alpha = 1$ reduces (10) to an IM update). In other words, the SOR update is given as $D_{SOR} = \alpha D_{IM}$. The pseudo-code for this acceleration technique is provided in Algorithm 2. The linesearch procedure, as its name implies, performs a line search along the given direction and will be discussed later in this section.

The second acceleration technique we investigated is the PARTAN method (e.g., see [17]). It was originally devised for accelerating gradient descent methods, in specific, to ameliorate the hallmark “zig-zag” sequence pattern of gradient-based updates, thus speeding up convergence. In our context, instead of applying it to gradients of $\sigma$, we use PARTAN directly on IM search directions given by (9). We illustrate the inner working of the PARTAN-accelerated IM update via Figure 1.

![PARTAN-IM UPDATE](image-url)
Algorithm 2 Successive Over-Relaxation acceleration of IM (SOR-IM)

**Input:** $W_0 \neq \mathcal{O}$, $t_{\text{max}} \geq 1$, $\tau > 0$, $s_{\text{max}} \leq 1$, $\alpha_{\text{init}} > 0$, $\eta_{\text{AT}} > 1$, $\eta_{\text{BT}} \in (0, 1)$, $c \in (0, 1)$

**Output:** $W_{\text{final}}$

1: for $t = 0$ to $t_{\text{max}}$ do
2: // Compute gradient matrix
3: $G_t \leftarrow [A - B(W_t)] W_t$
4: // Check for convergence
5: if $\|\text{vec}(G_t)\|_\infty \leq \tau$ then
6: $W_{\text{final}} \leftarrow W_t$
7: break
8: end if
9: // Compute IM’s search matrix
10: $D_{IM} \leftarrow -A^T G_t$
11: // Perform line search and update weights
12: $W_{t+1} \leftarrow \text{linesearch}(W_t, D_{IM}, G_t, \alpha_{\text{init}}, s_{\text{max}}, \eta_{\text{AT}}, \eta_{\text{BT}}, c)$
13: end for
14: $W_{\text{final}} \leftarrow W_{t_{\text{max}}}$
15: return $W_{\text{final}}$

Assume that $W_{t-1}$ and $W_t$ are generated by PARTAN-IM at iterations $t - 1$ and $t$ respectively. Next, our version of PARTAN seeks to find an intermediate point $W_{\text{interm}}$ along the IM search direction originating from $W_t$. We choose this update to coincide with an SOR update and, therefore, the step length will, in general, be greater than the one produced by IM. The PARTAN search direction for iteration $t + 1$ is given as $W_{\text{interm}} - W_{t-1}$, which is equivalent to

$$D_P(t + 1) = D_P(t) + D_{\text{SOR}}$$

(11)

In other words, PARTAN-IM modifies its previous search direction with an intermediate SOR step. Finally, the new iterate $W_{t+1}$ is found by the same line search procedure along PARTAN’s search direction. As long as two consecutive iterates of PARTAN-IM monotonically minimize the stress function (i.e. $\sigma(W_{t-1}) < \sigma(W_t)$), then it can be seen that the new iterate $W_{t+1}$ is guaranteed to produce an even lower stress function value. In the case, where PARTAN’s search direction is an ascent direction, one could revert to an SOR-IM update. However, we noticed in our experiments that this is rarely the case. Pseudo-code for PARTAN-IM is depicted in Algorithm 3.

In both SOR-IM and PARTAN-IM algorithms we used a specialized line search method, which consists of two parts. It starts with an initial step length of $\alpha_{\text{init}}$. If the update along search direction $D$ with the initial step length results in reduction of the stress function, then *ahead-tracking* is engaged to maximize this step-length. For every successful stress function value decrease, the step length is increased by a factor $\eta_{\text{AT}}$ until it is no more possible. On the other hand, if the update $\alpha_{\text{init}}D$ leads to an increase in stress function value, then *back-tracking* is employed in order to find a step length that satisfies Armijo’s condition [?] of sufficient decrease. For each unsuccessful step taken, the step length is decreased by a factor of $\eta_{\text{BT}}$. According to our experience, back-tracking is rarely employed in practice, but, still, it is necessary, so that the two acceleration techniques remain robust. Pseudo-code for our line search procedure is given in Algorithm 4.

Finally, we should note that $\alpha_{\text{init}} = 1$, $\eta_{\text{AT}} = 1.95$, $\eta_{\text{BT}} = 0.9$ and $c = 0.99$ are good, more or less, empirical values for the line search. We adopted these values to perform all of our experiments, that are presented in the next section, except for the Swiss Roll dataset, for which we obtained better results by using $\eta_{\text{AT}} = 2.0$. 

Algorithm 3 PARTAN acceleration of IM (PARTAN-IM)

**Input:** $W_0 \neq \mathcal{O}$, $t_{\text{max}} \geq 1$, $\tau > 0$, $s_{\text{max}} \leq 1$, $\alpha_{\text{init}} > 0$, $\eta_{\text{AT}} > 1$, $\eta_{\text{BT}} \in (0, 1)$, $c \in (0, 1)$

**Output:** $W_{\text{final}}$

1: // Compute initial gradient matrix
2: $G_0 \leftarrow [A - B(W_0)] W_0$
3: // Compute first iterate via IM
4: $W_1 \leftarrow A^T B(W_0) W_0$
5: for $t = 1$ to $t_{\text{max}}$ do
6: // Compute gradient matrix, if it is unavailable
7: $G_t \leftarrow [A - B(W_t)] W_t$
8: // Check for convergence
9: if $\|\text{vec}(G_t)\|_\infty \leq \tau$ then
10: $W_{\text{final}} \leftarrow W_t$
11: break
12: end if
13: // Compute IM’s search matrix
14: $D_{IM} \leftarrow -A^T G_t$
15: // Perform line search and compute intermediate matrix
16: $W_{\text{interm}} \leftarrow \text{linesearch}(W_t, D_{IM}, G_t, \alpha_{\text{init}}, s_{\text{max}}, \eta_{\text{AT}}, \eta_{\text{BT}}, c)$
17: // Compute PARTAN-IM’s search matrix
18: $D_P \leftarrow W_{\text{interm}} - W_{t-1}$
19: if $\text{trace}(D_P^T G_{t-1}) \leq 0$ then
20: // The search matrix corresponds to a descent direction; perform a PARTAN-IM step
21: $W_{t+1} \leftarrow \text{linesearch}(W_{t-1}, D_P, G_{t-1}, \alpha_{\text{init}}, s_{\text{max}}, \eta_{\text{AT}}, \eta_{\text{BT}}, c)$
22: else
23: // The search matrix corresponds to an ascent direction; perform an SOR-IM step
24: $G_{\text{interm}} \leftarrow [A - B(W_{\text{interm}})] W_{\text{interm}}$
25: $D_{IM} \leftarrow -A^T G_{\text{interm}}$
26: $W_{t+1} \leftarrow \text{linesearch}(W_{\text{interm}}, D_{IM}, G_{\text{interm}}, \alpha_{\text{init}}, s_{\text{max}}, \eta_{\text{AT}}, \eta_{\text{BT}}, c)$
27: $G_{t+1} \leftarrow G_{\text{interm}}$
28: end if
29: end for
30: $W_{\text{final}} \leftarrow W_{t_{\text{max}}}$
31: return $W_{\text{final}}$
Algorithm 4 linesearch() procedure employed by Algorithm 2 and Algorithm 3

Input: \( W_0 \neq O, \ D \neq O, \ G_0 \neq O, \ \alpha_{\text{init}} > 0, \ s_{\text{max}} \geq 1, \ \eta_{\text{AT}} > 1, \ \eta_{\text{BT}} \in (0, 1), \ c \in (0, 1), \ D \) must correspond to a descent direction

Output: \( W_{\text{next}} \)

1. \( \alpha \leftarrow \alpha_{\text{init}} \)
2. \( \sigma_0 \leftarrow \sigma(W_0) \)
3. \( W_1 \leftarrow W_0 + \alpha D \)
4. \( \sigma_1 \leftarrow \sigma(W_1) \)
5. if \( \sigma_1 < \sigma_0 \) then
   6. // Perform Ahead-Tracking
   7. \( \alpha \leftarrow \eta_{\text{AT}} \alpha \)
   8. for \( s = 1 \) to \( s_{\text{max}} \) do
   9. \( W_{s+1} \leftarrow W_s + \alpha D \)
10. \( \sigma_{s+1} \leftarrow \sigma(W_s) \)
11. if \( \sigma_{s+1} < \sigma_s \) then
   12. \( \alpha \leftarrow \eta_{\text{AT}} \alpha \)
   13. else
   14. break
   15. end if
16. end for
17. else
18. // Perform Back-Tracking
19. \( \Delta \sigma \leftarrow c \ \text{vec}^T (G_0) \ \text{vec}(D) \)
20. \( \alpha \leftarrow \eta_{\text{BT}} \alpha \)
21. for \( s = 1 \) to \( s_{\text{max}} \) do
22. \( W_{s+1} \leftarrow W_s + \alpha D \)
23. \( \sigma_{s+1} \leftarrow \sigma(W_s) \)
24. // Check sufficient decrease condition
25. if \( \sigma_{s+1} > \sigma_s + \alpha \Delta \sigma \) then
26. \( \alpha \leftarrow \eta_{\text{BT}} \alpha \)
27. else
28. break
29. end if
30. end for
31. end if
32. \( W_{\text{next}} \leftarrow W_s \)
33. return \( W_{\text{next}} \)

IV. EXPERIMENTAL RESULTS

A. Datasets

In order to evaluate the two acceleration schemes, we compare their convergence speed to the one of the original IM algorithm on five different data sets: 1) Open Box, 2) Teapots, 3) Swiss Roll, 4) Federalist Papers, 5) ORL Faces. For all experiments we used KSM with Gaussian kernels and \( H = N \) prototype vectors, that were randomly chosen from each data set. Inter-point Euclidean distances in the original space were used in place of dissimilarities. Furthermore, all data sets were projected onto the 2-dimensional plane, i.e. \( P = 2 \). The description of each data set follows, while their characteristics are summarized in Table I.

- **Open Box dataset.** It is an artificially-created data set that consists of points delineating an open box in 3 dimensions as shown in Figure 2a. KSM is used to project its samples onto the plane as demonstrated in Figure 2b. In order to test the scalability of the original IM algorithm and its two accelerated methods, we generated six versions of the data set with 10, 31, 64, 109, 166 and 235 training samples respectively.

- **Teapots dataset.** This dataset contains 100 different color images of the same artificially-rendered teapot under rotation every 3.6° [18], [19]. Each image consists of 560 × 420 pixels. After conversion to 8-bit grayscale, each image is represented as a 235200-dimensional vector. Since each image represents a 3.6° increment and, thus, only one degree of freedom is involved in this phenomenon, we attempted to project the images onto the plane as shown in Figure 3.

- **Swiss Roll dataset.** This artificially-generated data set contains samples of a rolled sheet in 3D space, as depicted in Figure 4a. The sheet’s surface is described by the \( \theta \)-parameterized equations \( x_1 = \theta \cos(\theta), \ x_2 = \theta \sin(\theta) \) and \( z \) is arbitrary. Since the intrinsic dimensionality of the manifold at hand is 2, we used KSM to project it onto the plane as shown in Figure 4b.

- **Federalist Papers dataset.** The Federalist Papers were written in 1787 and 1788 and published in many New
York newspapers to persuade the voters to ratify the US Constitution. Of all the collected papers, Alexander Hamilton wrote 56, James Madison wrote 50 and John Jay wrote 5 papers. There are 12 unidentified papers and most of people believe they are from Madison [20]. From this dataset we chose 14 training patterns from Hamilton, 14 from Madison and 12 unidentified papers. Each sample consists of a variety of features that try to capture information that could potentially identify a paper’s author, such as the writing style, vocabulary, etc. KSM was used to project these patterns onto the plane to depict (dis)similarities between papers from different authors. The result of this projection is given in Figure 5.

- **ORL Faces dataset.** This data set contains images of human faces taken between April 1992 and April 1994 at the Cambridge University Computer Laboratory [21]. There are 40 distinct subjects with each containing 10 different images at various poses, varying lighting conditions, facial expressions and facial details. Each image consists of $119 \times 92$ pixels. Again, we used KSM to visualize the relationships between the images of 3 different individuals by representing them as points in the plane. The projection result is illustrated in Figure 6.

In order to compare the three algorithms in terms of convergence speed, we ran each algorithm 100 times for each dataset using each time a different random initial weight matrix. Furthermore, for each run we recorded the number of iterations needed and the time (in seconds) it took for each algorithm to converge to a local minimum. We declared convergence, once the $L_\infty$ norm of the gradient reached or became lower than $\tau = 10^{-4}$. Finally, the values of the spread parameter $\psi$ employed for the Gaussian kernels were $800,000$ for Federalist Papers and $800,000$ for Swiss Roll, $8,000$ for Federalist Papers and $800$ for ORL Faces.

### B. Discussion

Our experimental results for the various versions of Open Box and the remaining datasets are summarized in Table II and Table III respectively. For Open Box I, in terms of median speed, SOR-IM is 57.52% faster than IM, while PARTAN-IM is 74.3% faster than IM. Next, for Open Box II, SOR-IM needs 38.07% less than IM, while PARTAN-IM is 50.01% faster than IM. For Open Box III, the relative difference in medians with respect to IM are 5.35% and 23.64% for SOR-IM and PARTAN-IM respectively. For Open Box IV, the differences are 9.06% between IM and SOR-IM and 34.14% between IM and PARTAN-IM. SOR-IM is 1.72% and PARTAN-IM is 13.49% faster than IM for Open Box.

![Fig. 4](image-url)  
**Fig. 4**  
The SwissRoll Dataset and its projection in 2D.

![Fig. 6](image-url)  
**Fig. 6**  
Result of Projection of the ORL Faces Dataset.

### Table I  
**Dataset Characteristics.**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of training patterns</th>
<th>Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Box I</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>Open Box II</td>
<td>34</td>
<td>3</td>
</tr>
<tr>
<td>Open Box III</td>
<td>64</td>
<td>3</td>
</tr>
<tr>
<td>Open Box IV</td>
<td>109</td>
<td>3</td>
</tr>
<tr>
<td>Open Box V</td>
<td>166</td>
<td>3</td>
</tr>
<tr>
<td>Open Box VI</td>
<td>235</td>
<td>3</td>
</tr>
<tr>
<td>Teapots</td>
<td>100</td>
<td>235200</td>
</tr>
<tr>
<td>Swiss Roll</td>
<td>120</td>
<td>3</td>
</tr>
<tr>
<td>Federalist Papers</td>
<td>40</td>
<td>18</td>
</tr>
<tr>
<td>ORL Faces</td>
<td>20</td>
<td>10304</td>
</tr>
</tbody>
</table>
### Table II
**Experimental Results for the Open Box Datasets.**

<table>
<thead>
<tr>
<th></th>
<th>IM time(sec)</th>
<th>Iterations</th>
<th>SOR-IM time(sec)</th>
<th>Iterations</th>
<th>PARTAN-IM time(sec)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Open Box I</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>1.406</td>
<td>168</td>
<td>0.453</td>
<td>60</td>
<td><strong>0.219</strong></td>
<td>32</td>
</tr>
<tr>
<td>Up 25</td>
<td>0.703</td>
<td>97.5</td>
<td>0.2965</td>
<td>40</td>
<td><strong>0.156</strong></td>
<td>22</td>
</tr>
<tr>
<td>Median</td>
<td>0.5155</td>
<td>71.5</td>
<td>0.219</td>
<td>31</td>
<td><strong>0.1325</strong></td>
<td>19</td>
</tr>
<tr>
<td>Low 25</td>
<td>0.375</td>
<td>53</td>
<td>0.172</td>
<td>23</td>
<td><strong>0.109</strong></td>
<td>16.5</td>
</tr>
<tr>
<td>Min</td>
<td>0.281</td>
<td>40</td>
<td>0.094</td>
<td>15</td>
<td><strong>0.078</strong></td>
<td>14</td>
</tr>
<tr>
<td><strong>Open Box II</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>3.063</td>
<td>573</td>
<td>1.719</td>
<td>149</td>
<td><strong>0.859</strong></td>
<td>52</td>
</tr>
<tr>
<td>Up 25</td>
<td>0.945</td>
<td>153.5</td>
<td>0.594</td>
<td>56</td>
<td><strong>0.461</strong></td>
<td>33</td>
</tr>
<tr>
<td>Median</td>
<td>0.7815</td>
<td>131.5</td>
<td>0.484</td>
<td>47</td>
<td><strong>0.39</strong></td>
<td>29</td>
</tr>
<tr>
<td>Low 25</td>
<td>0.688</td>
<td>122</td>
<td>0.437</td>
<td>41</td>
<td><strong>0.344</strong></td>
<td>26.5</td>
</tr>
<tr>
<td>Min</td>
<td>0.438</td>
<td>86</td>
<td>0.312</td>
<td>30</td>
<td><strong>0.282</strong></td>
<td>23</td>
</tr>
<tr>
<td><strong>Open Box III</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>9.781</td>
<td>850</td>
<td>8.843</td>
<td>260</td>
<td><strong>4.203</strong></td>
<td>82</td>
</tr>
<tr>
<td>Up 25</td>
<td>2.812</td>
<td>244.5</td>
<td>2.64</td>
<td>79.5</td>
<td><strong>2.1955</strong></td>
<td>45</td>
</tr>
<tr>
<td>Median</td>
<td>2.4765</td>
<td>217.5</td>
<td>2.344</td>
<td>72</td>
<td><strong>1.891</strong></td>
<td>39</td>
</tr>
<tr>
<td>Low 25</td>
<td>1.9295</td>
<td>168</td>
<td>1.969</td>
<td>60.5</td>
<td><strong>1.7185</strong></td>
<td>36</td>
</tr>
<tr>
<td>Min</td>
<td>1.438</td>
<td>128</td>
<td>1.453</td>
<td>46</td>
<td><strong>1.344</strong></td>
<td>29</td>
</tr>
<tr>
<td><strong>Open Box IV</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>20.6870</td>
<td>695</td>
<td>20.4210</td>
<td>232</td>
<td><strong>13.2040</strong></td>
<td>97</td>
</tr>
<tr>
<td>Up 25</td>
<td>15.2180</td>
<td>518.5</td>
<td>14.1250</td>
<td>158.5</td>
<td><strong>9.6090</strong></td>
<td>71.5</td>
</tr>
<tr>
<td>Median</td>
<td>12.5160</td>
<td>420</td>
<td>11.3825</td>
<td>84</td>
<td><strong>8.2425</strong></td>
<td>63.5</td>
</tr>
<tr>
<td>Low 25</td>
<td>10.6565</td>
<td>370.5</td>
<td>9.6175</td>
<td>110.5</td>
<td><strong>6.7895</strong></td>
<td>52.5</td>
</tr>
<tr>
<td>Min</td>
<td>7.4370</td>
<td>264</td>
<td>6.8130</td>
<td>82</td>
<td><strong>5.0000</strong></td>
<td>40</td>
</tr>
<tr>
<td><strong>Open Box V</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>20.3750</td>
<td>472</td>
<td>20.2170</td>
<td>142</td>
<td><strong>12.5400</strong></td>
<td>97</td>
</tr>
<tr>
<td>Up 25</td>
<td>16.6300</td>
<td>335.5</td>
<td>15.5900</td>
<td>103.5</td>
<td><strong>16.5630</strong></td>
<td>57.5</td>
</tr>
<tr>
<td>Median</td>
<td>15.3867</td>
<td>258.5</td>
<td>14.5940</td>
<td>84</td>
<td><strong>13.7265</strong></td>
<td>48</td>
</tr>
<tr>
<td>Low 25</td>
<td>13.7340</td>
<td>227</td>
<td>13.5705</td>
<td>72.5</td>
<td><strong>12.0630</strong></td>
<td>43</td>
</tr>
<tr>
<td>Min</td>
<td>11.3130</td>
<td>186</td>
<td>10.8120</td>
<td>58</td>
<td><strong>10.2970</strong></td>
<td>37</td>
</tr>
<tr>
<td><strong>Open Box VI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>91.9380</td>
<td>720</td>
<td>74.4840</td>
<td>187</td>
<td><strong>57.3440</strong></td>
<td>96</td>
</tr>
<tr>
<td>Up 25</td>
<td>56.0240</td>
<td>438</td>
<td>54.7500</td>
<td>137</td>
<td><strong>41.7500</strong></td>
<td>71</td>
</tr>
<tr>
<td>Median</td>
<td>50.1645</td>
<td>391.5</td>
<td>48.9840</td>
<td>123</td>
<td><strong>37.0395</strong></td>
<td>63</td>
</tr>
<tr>
<td>Low 25</td>
<td>37.9525</td>
<td>298</td>
<td>38.3435</td>
<td>97</td>
<td><strong>31.5155</strong></td>
<td>55</td>
</tr>
<tr>
<td>Min</td>
<td>24.0780</td>
<td>189</td>
<td>27.5630</td>
<td>71</td>
<td><strong>22.6410</strong></td>
<td>40</td>
</tr>
</tbody>
</table>

### Table III
**Experimental Results on the remaining datasets.**

<table>
<thead>
<tr>
<th></th>
<th>IM time(sec)</th>
<th>Iterations</th>
<th>SOR-IM time(sec)</th>
<th>Iterations</th>
<th>PARTAN-IM time(sec)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teapots Max</td>
<td>101.609</td>
<td>4270</td>
<td>39.015</td>
<td>475</td>
<td><strong>31.656</strong></td>
<td>278</td>
</tr>
<tr>
<td>Up 25</td>
<td>44.43</td>
<td>1844.5</td>
<td>21.0075</td>
<td>249</td>
<td><strong>16.8125</strong></td>
<td>148</td>
</tr>
<tr>
<td>Median</td>
<td>30.25</td>
<td>1265.5</td>
<td>19.8200</td>
<td>103.5</td>
<td><strong>16.5630</strong></td>
<td>57.5</td>
</tr>
<tr>
<td>Low 25</td>
<td>20.1015</td>
<td>1001.5</td>
<td>12.4375</td>
<td>92</td>
<td><strong>10.8205</strong></td>
<td>97.5</td>
</tr>
<tr>
<td>Min</td>
<td>11.828</td>
<td>502</td>
<td>7.469</td>
<td>92</td>
<td><strong>6.656</strong></td>
<td>59</td>
</tr>
<tr>
<td>Swiss Max</td>
<td>52.5530</td>
<td>1602</td>
<td>27.0010</td>
<td>187</td>
<td><strong>19.3100</strong></td>
<td>104</td>
</tr>
<tr>
<td>Roll Up 25</td>
<td>16.6300</td>
<td>402</td>
<td>15.5495</td>
<td>125.5</td>
<td><strong>12.2615</strong></td>
<td>71</td>
</tr>
<tr>
<td>Median</td>
<td>12.8095</td>
<td>321</td>
<td>12.3500</td>
<td>103.5</td>
<td><strong>9.8500</strong></td>
<td>62.5</td>
</tr>
<tr>
<td>Low 25</td>
<td>10.1860</td>
<td>249.5</td>
<td>9.8535</td>
<td>84</td>
<td><strong>8.3155</strong></td>
<td>55</td>
</tr>
<tr>
<td>Min</td>
<td>4.3780</td>
<td>137</td>
<td>6.3830</td>
<td>59</td>
<td><strong>5.6740</strong></td>
<td>35</td>
</tr>
<tr>
<td>Federalist Max</td>
<td>9.406</td>
<td>1482</td>
<td>3.484</td>
<td>226</td>
<td><strong>2.172</strong></td>
<td>100</td>
</tr>
<tr>
<td>Papers Up 25</td>
<td>2.3905</td>
<td>439.5</td>
<td>1.906</td>
<td>125</td>
<td><strong>1.407</strong></td>
<td>66.5</td>
</tr>
<tr>
<td>Median</td>
<td>1.906</td>
<td>361.5</td>
<td>1.6015</td>
<td>105</td>
<td><strong>1.1955</strong></td>
<td>56</td>
</tr>
<tr>
<td>Low 25</td>
<td>1.492</td>
<td>272.5</td>
<td>1.266</td>
<td>82</td>
<td><strong>1</strong></td>
<td>48</td>
</tr>
<tr>
<td>Min</td>
<td>0.844</td>
<td>176</td>
<td>0.813</td>
<td>55</td>
<td><strong>0.703</strong></td>
<td>36</td>
</tr>
<tr>
<td>ORL Max</td>
<td>3.359</td>
<td>1222</td>
<td>0.75</td>
<td>97</td>
<td><strong>0.515</strong></td>
<td>72</td>
</tr>
<tr>
<td>Faces Up 25</td>
<td>1.4295</td>
<td>231</td>
<td>0.391</td>
<td>54</td>
<td><strong>0.25</strong></td>
<td>35.5</td>
</tr>
<tr>
<td>Median</td>
<td>0.6485</td>
<td>153</td>
<td>0.297</td>
<td>44</td>
<td><strong>0.203</strong></td>
<td>28.5</td>
</tr>
<tr>
<td>Low 25</td>
<td>0.0445</td>
<td>84</td>
<td>0.204</td>
<td>36</td>
<td><strong>0.1485</strong></td>
<td>23</td>
</tr>
<tr>
<td>Min</td>
<td>0.094</td>
<td>53</td>
<td>0.109</td>
<td>21</td>
<td><strong>0.079</strong></td>
<td>17</td>
</tr>
</tbody>
</table>
V. Finally, for Open Box VI, SOR-IM and PARTAN-IM are 2.35% and 26.16% faster respectively than IM. These results, which are also summarized in Figure 8, indicate that SOR-IM’s performance advantage in comparison to IM is probably eroding, as the number of weights increase. A similar conclusion can probably be drawn for PARTAN-IM as well. However, the performance deterioration is much slower for the latter algorithm. Box plots of the execution time distributions for each algorithm on selected Open Box datasets are given in Figure 7.

For the Teapots data set, Table III reflects that median SOR-IM speed performance is 45.95% faster than the corresponding IM performance. On the other hand, PARTAN-IM has 55.48% faster median performance than IM. For Swiss Roll, the SOR-IM median slightly outperforms IM’s by 3.72%, but PARTAN-IM is still 23.04% faster than IM. Next, for the Federalist Papers dataset, median SOR-IM performance is 15.98% less than the one of IM, while PARTAN-IM is faster by 37.28%. As for ORL Faces, median speedups for SOR-IM and PARTAN-IM are 54.2% and 68.7% respectively faster than IM’s median performance. Box plots of the execution time distributions for the Teapots and Swiss Roll data sets are given in Figure 9. Similar Box plots are obtained for the other two remaining data sets as well, but are not shown here.

V. Conclusions

We explored 2 acceleration methods, namely SOR Accelerated IM (SOR-IM) and PARTAN Accelerated IM (PARTAN-IM), to improve the convergence speed of the Iterative Majorization (IM) algorithm, which is used to estimate the projection weights of a Generalized Sammon Mapping (GSM) model. Both methods are relatively easy to understand and to implement. Based on our experiences, we can conclude that both methods indeed accelerate the original algorithm and exhibit robust behavior. We have shown experimentally, that PARTAN-IM always converges faster than SOR-IM, which, in turn, is typically faster than IM. This is despite the relatively increased complexity of these algorithms over IM. However, it seems that their speedup advantage over plain IM seems to decrease as the number of weights increase, a fact that is more or less expected. Nevertheless, they may still retain an advantage, albeit much smaller, even as the size of the dataset to be projected increases.

Acknowledgment

Yinjie Huang acknowledges partial support from NSF grants No. 0647120 and No. 0717680 and partial support from a UCF Graduate College Presidential Fellowship. Moreover, Michael Georgiopoulos acknowledges partial support from NSF grants No. 0525429, No. 0837332 and No. 0717680. Finally, Georgios C. Anagnostopoulos acknowledges partial support from NSF grants No. 0717674 and No. 0647018. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. Finally, the authors would like to thank the anonymous reviewers of this manuscript for their time and helpful comments.
REFERENCES


