An accurate MDS-based algorithm for the visualization of large multidimensional datasets

Antoine Naud

Department of Informatics, Nicolaus Copernicus University, Toruń, Poland,
naud@phys.uni.torun.pl,
http://www.phys.uni.torun.pl/~naud

Abstract. A common task in data mining is the visualization of multivariate objects on scatterplots, allowing human observers to perceive subtle inter-relations in the dataset such as outliers, groupings or other regularities. Least- squares multidimensional scaling (MDS) is a well known Exploratory Data Analysis family of techniques that produce dissimilarity or distance preserving layouts in a non-linear way. In this framework, the issue of visualizing large multidimensional datasets through MDS-based methods is addressed. An original scheme providing very accurate layouts of large datasets is introduced. It is a compromise between the computational complexity $O(N^{5/2})$ and the accuracy of the solution that makes it suitable both for visualization of fairly large datasets and preprocessing in pattern recognition tasks.

1 Introduction

The increasing amount of data available over the Internet gives rise to a need in efficient data analysis tools allowing an easier use of large databases. Data visualization is often a necessary step in a data analysis process because it permits to detect the presence of clusters or other regularities in data. This paper focuses on dimensionality reduction methods as tools for the visualization of large multidimensional datasets, as well as a feature extraction of such data. These tasks have been successfully performed by neural networks as the Self-Organizing Maps [12], or by kernel methods [16], latent variable methods as the GTM [8] or multidimensional scaling (MDS) [20]. In order to improve the quality of layouts and to adapt them to the visualization of increasingly growing datasets, newly developed approaches to the above models include local dimensionality reduction and hierarchical visualization.

The visualization of large datasets using full scaling is often unpractical due to the algorithmic complexity $O(N^3)$, where N is the number objects simultaneously mapped. Such applications are limited to a few thousands items sized datasets. A strategy to alleviate this constraint is to split the dimensionality reduction process into two steps: first a smaller dataset built from the data (obtained by clustering or any other method) is mapped, and second the input data is added in some way to the smaller dataset's layout obtained in the first step.

This general scheme for large scale dimensionality reduction has been realized in many ways, using various approaches for the construction of the smaller dataset and for the choice of the reduction technique. We mention here below some approaches very

closely related to our MDS-based proposal. Basalaj proposed *incremental scaling* [1] where data points are incrementally added through a single (points are added 1 by 1) least-squares scaling. The order in which points are added is extracted from a MST of the data. This scheme leads to $O(N^{7/3})$ complexity. Brodbeck and Girardin [2] use the clustering capability of SOM and a spring model to produce whether local layouts of cluster neighborhoods, or one global layout of the cluster centers. Morrison et al. [3] [4] [5] use a sample of $\sqrt{N}$ items instead of a data clustering followed by an interpolation strategy also proposed by Brodbeck and Girardin, achieving very low complexities: $O(N^2)$, $O(N\sqrt{N})$ and $O(N^{5/4})$, allowing to visualize a dataset of 108,000 14-dimensional objects. Schwenker et al. [6] combine in ACMDS adaptive c-means and classical scaling. Williams and Muntzer [17] designed a steerable and progressive MDS capable of visualizing 120,000 items and 294 dimensions in a few hours, using hierarchical structures to select subsets of interest and progressive, in-depth and localized layouts. There are also many algorithms proposed to adapt linear dimensionality reduction algorithms such as classical scaling to the visualization of large datasets, let us mention among others FastMap [19] and Locally Linear Embeding [10].

Our approach is to first build the smaller dataset (called Basis) using a k-means clustering of the input data and map it using standard least-squares MDS. Then input data is added to the Basis layout using *relative MDS* [14]. This new association scheme of k-means clustering and multidimensional scaling is introduced in next Section 2. In Section 3, experiments on 3 real datasets show the validity of the proposed scheme. A short conclusion summarizes this paper.

2 A new approach to the association of MDS to k-means clustering

In least-squares MDS, the preservation of neighborhood relationships is ensured by the minimization of the Stress functional $S(\mathbf{Y})$ defined as

$$S(\mathbf{Y}) = \frac{1}{F_n} \sum_{i<j}^{N} w_{ij} \cdot (D_{ij} - d_{ij}(\mathbf{Y}))^2 \tag{1}$$

in which $\mathbf{Y}$ is the matrix of coordinates of N points representing the given N D-dimensional objects in the output d-dimensional space. $\{D_{ij}\}$ are given dissimilarities or the inter-object distances, $\{w_{ij}\}$ are weighting factors that permit to tune the impact of large distance on the sum (hence w_{ij} is generally inversely proportional to D_{ij}), and $\{d_{ij}\}$ are the output space inter-point distances. F_n is a normalization factor to keep Stress values in unit range $[0, 1]$. The minimization of functional $S(\mathbf{Y})$ with respect to the $N \times d$ variables can be realized in various manners that may be local or global optimization, with more or less accurate and time consuming procedures. In our implementation, a steepest descent procedure is used, including second order derivatives in such a way that it is not as computationally intensive as a real Newton method. It was found to be a good compromise between accuracy of the solution and computational complexity.

In a first step, a N_B-sized Basis is build from the set of cluster centers obtained by a standard k-means clustering. Other cluster algorithms have been tested in this

framework [15] (Learning Vector Quantization or dendrograms), but it appeared that k-means clustering is best suited to this task (i.e. leading to layouts with lower final Stress values). This result was confirmed by an experiment on one dataset, in which 100 random Bases were generated as N_B-sized samples of the Data, and the final layouts resulting from their use were compared in terms of the general Stress expression (1). The best final layout was obtained for the Basis made of points very close to the k-means cluster centers. The association scheme presented in this paper is similar to the one presented in [15], but is gives much better results (lower stresses) at relatively small cost. The first step is identical: we map the N_B Basis points using standard least-squares MDS, that is minimizing the *Stress* functional $S_b(\mathbf{Y})$ defined as

$$S_b(\mathbf{Y}) = \frac{1}{F_n} \sum_{i<j}^{N_B} w_{ij} \cdot (D_{ij} - d_{ij}(\mathbf{Y}))^2 \,, \tag{2}$$

The difference from the scheme of paper [15] lies in the second step, where input data is not added on a point by point basis (as Basalaj did), but into K batches of N_C input data ($N_C = N_B$, except for the last batch in which $N_C \leq N_B$, and $K = \lceil N/N_C \rceil$). In each batch only a subset of N_C input data is added to the Basis layout by relative MDS. So this step consists in the minimizations of a series of Stress functionals defined as

$$S_{r,k}(\mathbf{Y}) = \frac{1}{F_n} \sum_{i<j}^{N_C} w_{ij} \cdot (D_{ij} - d_{ij}(\mathbf{Y}))^2 + \frac{1}{F_n} \sum_{i=1}^{N_C} \sum_{j=1}^{N_B} w_{ij} \cdot (D_{ij} - d_{ij}(\mathbf{Y}))^2 \,, \tag{3}$$

for $k = 1, \cdots, K$. The layout resulting of this relative MDS mapping scheme cannot be as low as the one obtained by one full MDS of the entire dataset, because in relative MDS distances between points added in separate batches are never taken into account. For this reason, adding points in batches of small subsets, whose inter-points distances are included in Stress expression (3) should give better results than adding points one by one. We have at hand groupings from the clustering stage: the cluster centers neighborhoods (the set of points whose a center is the closest). Expressions (2) and (3) give inherently more weight to larger distances, and even more when squared distances are used. This suggests to form groups by picking up one point from each cluster center neighborhood, in order to force having as much large distances as possible in each relative mapping batch, to finally produce to a lower Stress. We call this relative mapping using *inter-cluster groups*. The experiments presented in next section will show that this intuitive approach is well-founded.

To reduce the computation time, the above Stress expressions have been simplified by using squared Stress (*SStress*) where all input and output distances are squared Euclidean distances. Besides this, all the weights $\{w_{ij}\}$ are taken equal to 1 and the normalization factor is $F_n = \sum_{i<j}^{N} D_{ij}^2$, which leads to a Stress functional called here below SS_1. The stopping criterion for the minimizations iterative process was the gradient length per variable, that is divided by $N_m \times d$, ($N_m = \{N_B, N_C\}$ is the number of mapped points and d is the dimensionality of the output space). The stopping threshold value was $\epsilon_G = 1.0E-12$ for full MDS and $\epsilon_G = 1.0E-8$ for relative MDS, which yields in general to a number of iterations of the same order as N_m.

As proven by expressions (4-6) and shown in next section's experiments, minimizing expressions (2) and (3) is faster than minimizing the original expression (1). We consider that the number of iterations needed in one standard MDS minimization process is proportional to N, whereas it is proportional to $\sqrt{N}$ in relative MDS. Computational complexities CC_{full} of full MDS on input data, CC_{single} of relative MDS in single batches and CC_{groups} of relative MDS of N_C-sized batches can be assessed as follows (following Chalmers et al. [4], we set $N_B = \sqrt{N}$ to simplify the expressions, and neglecting the clustering stage):

$$CC_{full} = O\left(N^3\right), \tag{4}$$

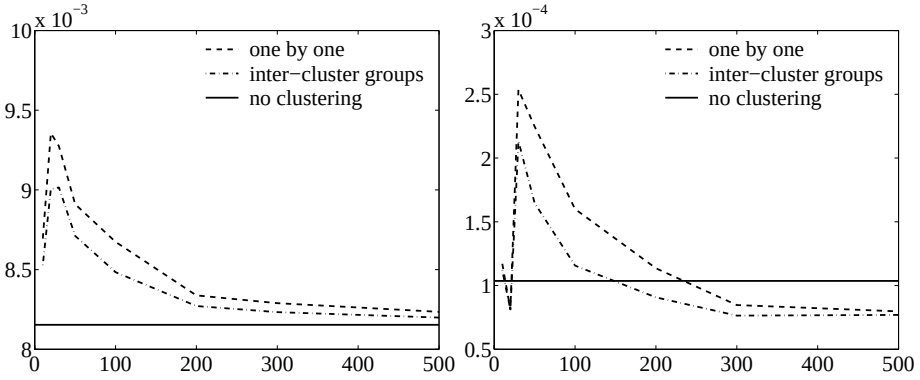
$$CC_{single} = CC_{step1} + CC_{step2} = O\left(N_b^3\right) + O\left(N_b^2 N\right) \approx O\left(N^2\right), \tag{5}$$

$$CC_{groups} = CC_{step1} + CC_{step2} = O\left(N_b^3\right) + O\left((N/N_b)^2 N_b N\right) \approx O\left(N^{5/2}\right), \tag{6}$$

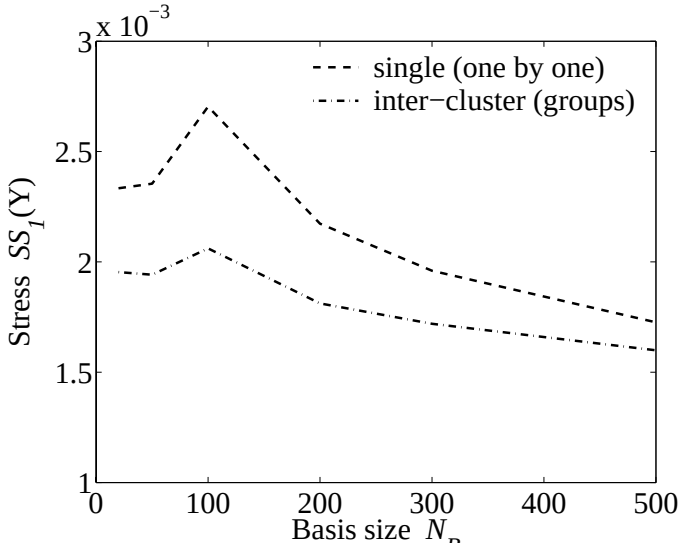
3 Experiments on real datasets

We tested our approach on two well known real datasets, namely `satimage` and `abalone` from the UCI repository [9] for the following reason: In order to assess the accuracy of the results obtained by our approach, we need to compare their Stress values to the ones obtained by full scaling of the entire datasets. Those datasets are similar in size (4435 items with 36 features in `satimage`, and 4177 items with 7 numerical features in `abalone`), which allows full scaling of the entire datasets because of reasonable time and memory requirements. The different relative MDS mappings performances are the final Stress values from expr. (1) for the whole datasets and they will be compared to what should be their optimal values (obtained by direct MDS mapping of the entire datasets). In order to evaluate its scalability, the method was also applied to a larger dataset called here `texture`, it is a fragment of the `Corel Image Features` dataset from the UCI KDD Archive [9] (from the 4 sets of features available, we took the co-occurence texture with 16 features and 68040 items).

The *k-means* clustering used is the compiled standard k-means iterative approach from Matlab. Since the k-means and relative MDS algorithms are not deterministic, they were run 20 times for each mapping, keeping only the best solution. Although k-means is considered as a fast clustering method, its application occurred to be the bottleneck of our process for larger Basis sizes, i.e. when $N_B > 100$, and making prohibitive Basis of size $N_B > 500$. The resulting Stress values are presented on Fig. 3. As could be expected, the Stress decreases when N_B increases, due to the increasing number of Basis reference points allowing a more precise location of the added data. For smaller N_B values, we observe important Stress variations probably related to the *curse of dimensionality* occurring as the number of objects is too law w.r.t. the number of dimensions of the input data. The bottom solid lines represent the minimal Stress value reached using full MDS on the whole dataset: $SS_1 = 1.04E-04$ for `abalone` dataset and $SS_1 = 8.15E-03$ for `satimage` dataset. It is interesting to note that the minimum reached by full MDS for `abalone` dataset is outperformed by relative MDS with Basis sizes $N_B > 200$. This result shows that relative MDS does not only



(a) `satimage` dataset (b) `abalone` dataset



(c) `texture` dataset

Fig. 1. Final Stress values obtained by 3 different MDS-based mapping methods, for varying Basis size $N_B \in [10, 500]$. The bottom dotted line shows the minimum reached in one MDS mapping of the entire dataset without clustering. The performance superiority of inter-cluster groups mapping is clearly visible in each case, especially for $N_B \in [100, 200]$.

provide faster mappings for large datasets than standard MDS, but it can also reach *better* solutions.

Execution times for those 3 datasets are presented in Table 3. Full MDS was not performed on the entire `texture` dataset for obvious prohibitive time and memory requirements. Full MDS and the different relative MDS versions were all run 20 times. The relative MDS runs were for $N_B = 100$. The two right-most columns present execution times for relative MDS using batches formed directly by cluster centers neighborhoods (groups1) and by batches formed by picking randomly one input data from each neighborhood (groups2). The durations differences between the datasets for relative MDS can vary from one Basis size to another. These performances can be reduced if we decrease the iterations stopping criterion ϵ_G, at the cost of less accurate final layouts.

Dataset	full MDS	Relative MDS			
	(Data)	(Basis)	(Single)	(Groups1)	(Groups2)
`abalone`	65500	52	374	697	3042
`satimage`	60196	5.6	43	65	210
`texture`	–	34	1448	7283	7372

Table 1. Execution times in seconds on an Intel Celeron CPU 2.2 GHz for the 3 datasets. The computations for full MDS mappings (left-most column) were run on a Pentium IV CPU 3.0GHz.

4 Conclusion

This paper presents a new way to combine k-means clustering and multidimensional scaling, as an alternative to other approaches reducing the computational complexity of multidimensional scaling. The proposed association of relative MDS scaling allowed to obtain accurate layouts of datasets of size up to 68000 items in a two hours. The computational complexity of the designed process is reduced by combining MDS to a naive iterative k-means clustering. The resulting solutions present very good Stress performances, sometimes even outperforming the results of full MDS solutions. Experiments showed that the bottleneck of the whole process as it is implemented now is the k-means clustering. More efficient clustering techniques should be used in the future such as the ones proposed in [11] [7] [21] in order to speed up the clustering stage. The proposed scheme can be applied not only to data visualization, but everywhere a dimensionality reduction of data is needed, for instance as a preprocessing stage in pattern recognition applications.

References

1. W. Basalaj “Incremental multidimensional scaling method for database visualization,” *Proceedings of the Visual Data Exploration and Analysis VI*, SPIE, vol. 3643, pp. 149–158, 1999.

2. Brodbeck, D., L. Girardin, “Combining Topological Clustering and Multidimensional Scaling for Visualising Large Data Sets”, Unpublished paper (accepted for, but not published in *Proceedings of the IEEE Information Visualization 1998*)
3. M. Chalmers “A linear iteration time layout algorithm for visualising high-dimensional data,” *Proceedings of the IEEE Visualization’96* , San Francisco, pp. 127-132, Oct.-Nov. 1996.
4. Morrison A., Ross G., Chalmers M. “Fast multidimensional scaling through sampling, springs and interpolation”, *Proceedings of the Information Visualization 2*, 1, pp. 68Ű77, 2003.
5. Morrison A., Chalmers M. “Improving hybrid MDS with pivot-based searching”, *Proceedings of the Information Visualization 4*, 2, pp. 109Ű122, 2005.
6. F. Schwenker, H. Kestler and G. Palm “Algorithms for the visualization of large and multivariate datasets,” in *Self-organizing neural networks* U. Seiffet and L. C. Jain eds, chap. 8 pp. 165-183, Physica-Verlag, Heidelberg, 2002.
7. D. Pelleg and A. Moore, “Accelerating Exact k -means Algorithms with Geometric Reasoning,” in "Knowledge Discovery and Data Mining", pp. 277-281, 1999.
8. C.M. Bishop, J.F.M. Svensen and C.K.I. Williams “GTM: The Generative Topographic Mapping,” *Neural Computation* vol. 10(1), pp. 215–234, Jan. 1998.
9. C.L Blake and C.J. Merz, “UCI Repository of machine learning databases,” Irvine, CA: University of California, Department of Information and Computer Science, 1998.
10. L. K. Saul and S. T. Roweis “Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds,” Journal of Machine Learning Research, 4, pp. 119-155, 2003.
11. Kanungo and Mount *A local search approximation algorithm for k-means clustering* Heidelberg, Berlin: Springer-Verlag, 1995.
12. T. Kohonen *Self-Organizing Maps* Heidelberg, Berlin: Springer-Verlag, 1995.
13. A. Naud and W. Duch “Interactive data exploration using MDS mapping,” *Proceedings of the Fifth Conference on Neural Networks and Soft Computing*, Zakopane, pp. 255–260, 2000.
14. A. Naud and W. Duch “Visualization of large datasets using MDS combined with LVQ” *Proceedings of the Sixth International Conference on Neural Networks and Soft Computing*, Zakopane 2002, pp. 632–637, L. Rutkowski and J. Kacprzyk eds.
15. A. Naud “Visualization of high-dimensional data using an association of multidimensional scaling to clustering” *Proceedings of the 2004 IEEE Cybernetics and Intelligent Systems*, Singapore 2004.
16. B. Schölkopf, A. Smola and K.-R. Müller “Nonlinear Component analysis as a Kernel Eigenvalue Problem,” *Neural Computation*, vol. 10(5), pp. 1299–1319, July ‘1998.
17. M. Williams and T. Munzner “Steerable, Progressive Multidimensional Scaling,” *Proceedings of the InfoVis 2004*, pp. 57-64, 2004.
18. K. Alsabti, S. Ranka, and V. Singh. “An efficient k-means clustering algorithm,” *Proceedings of the IPPS/SPDP Workshop on High Performance Data Mining*, 1998.
19. Christos Faloutsos and King-Ip Lin, “FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets,” *Proceedings of the SIGMOD Conference*, pp. 163-174, 1995.
20. T. F. Cox and M. A.A. Cox “Multidimensional Scaling,”, Monographs on Statistics and Applied Probability, vol. 59, Chapman & Hall, 1994.
21. Tapas Kanungo and David M. Mount and Nathan S. Netanyahu and Christine D. Piatko and Ruth Silverman and Angela Y. Wu “An Efficient k-Means Clustering Algorithm: Analysis and Implementation,” in IEEE Trans. PAMI, 24 (7), pp. 881-892, 2002.