INCREMENTAL VISUALIZATION OF GROWING MUSIC COLLECTIONS

Sebastian Stober, Thomas Low, Tatiana Gossen & Andreas Nürnberger
Data & Knowledge Engineering Group, Faculty of Computer Science, University of Magdeburg, DE
{stober, thomas.low, tatiana.gossen, andreas.nuernberger}@ovgu.de

ABSTRACT

Map-based visualizations – sometimes also called projections – are a popular means for exploring music collections. But how useful are they if the collection is not static but grows over time? Ideally, a map that a user is already familiar with should be altered as little as possible and only as much as necessary to reflect the changes of the underlying collection. This paper demonstrates to what extent existing approaches are able to incrementally integrate new songs into existing maps and discusses their technical limitations. To this end, Growing Self-Organizing Maps, (Landmark) Multidimensional Scaling, Stochastic Neighbor Embedding, and the Neighbor Retrieval Visualizer are considered. The different algorithms are experimentally compared based on objective quality measurements as well as in a user study with an interactive user interface. In the experiments, the well-known Beatles corpus comprising the 180 songs from the twelve official albums is used – adding one album at a time to the collection.

1. INTRODUCTION

Within the last decade, two-dimensional maps have become a popular means for visualizing music collections. By providing a collection overview which easily allows to identify regions or neighborhoods of similar songs, they are especially helpful if users want to explore a collection without having to formulate an explicit query. There exists a large variety of approaches to compute maps. The most popular ones in the field of Music Information Retrieval (MIR) are Self-Organizing Maps (SOMs) [14] and Multidimensional Scaling (MDS) [15] as well as related techniques such as Principle Component Analysis (PCA) [10]. Considerable effort has been made to improve the quality and usefulness of the generated maps – e.g., by adapting the underlying similarity measure based on user feedback [11, 27], enriching the visualization with landscape features such as islands [5, 13, 21, 22, 24] and mountain ranges [18, 20], or correcting projection errors (caused by the inherent dimensionality reduction during map generation) with an adaptive distortion technique [29].

However, to the knowledge of the authors, the problem of changing collections has not yet been addressed appropriately. To what extent does a map change when songs are added or removed? Could it even be necessary to recompute the map from scratch? Such questions need to be answered as failing to support collection changes may significantly limit the usefulness of a MIR application in real-world scenarios. Here, being able to add songs to an existing map is more important than removal which is often trivial (at worst leading to blank spaces in the map) and an uncommon use case anyways. New songs that are similar to existing ones should be embedded in the respective neighborhoods. At the same time, the map should also be able to deal with changes in music taste (adding songs from new genres) and an increase of musical diversity. Ideally, a map should be altered as little as possible and only as much as necessary to reflect the changes of the underlying collection. Too abrupt changes in the topology might confuse the user who over time will get used to the location of specific regions in the map.

In this paper, we compare several popular approaches for generating two-dimensional maps of music collections with respect to their ability to deal with growing collections. Section 2 briefly reviews the algorithms covered in the comparison and points out related work. The evaluation has been two-fold: We conducted objective measurements (Section 3) as well as a user study (Section 4). Section 5 summarizes our findings and draws conclusions.

2. LAYOUT ALGORITHMS

2.1 Multidimensional Scaling (MDS)

Given a set of data points, classical MDS [15] finds an embedding in the target space (here $\mathbb{R}^2$) that maintains their distances (or dissimilarities) as far as possible – without having to know their actual values. This way, it is also well suited to compute a layout for spring- or force-based approaches or as a method for vectorization (Section 2.3). MDS is closely related to PCA [33], which projects data points simply onto the (two) axes of highest variance termed principal components. In contrast to SOMs, both are non-parametric approaches that compute an optimal solution (with respect to data variance maximization and distance preservation respectively) in fixed polynomial time. Systems that apply PCA, MDS or similar force-based approaches comprise [1, 6, 31], the *fm4 Soundpark* [5], *MusicBox* [16], and *SoundBite* [17].

MDS does not support incremental collection changes.

Instead, a new map has to be computed every time the collection grows. Even with little change of the collection, the resulting map may look very different because it could be arbitrarily translated, rotated, and reflected without affecting the pairwise distances. In order to remedy this issue, we apply Procrustes analysis [7] to align each newly generated map with the previous one. This method allows to transform a set of data points through translation, rotation, and uniformly scaling such that it resembles another set as closely as possible.

2.2 Landmark Multidimensional Scaling

Landmark MDS as described in [2] is a computationally efficient approximation to classical MDS. The general idea of this approach is as follows: Given a sample set of landmark or pivot objects, an embedding into a low-dimensional space is computed for these objects using classical MDS. Each remaining object can then be located within the output space according to its distances to the landmarks. Obviously, the quality of the projection depends on the choice of the landmarks – especially if the landmark sample set is small compared to the size of the whole dataset. If the landmarks lie close to a low-dimensional subspace (e.g., a line), there is the chance of systematic errors in the projection.

As described in [29], Landmark MDS can be applied to visualize growing music collections by using the initial songs as landmarks. This way, the position of a song once added to the map never changes. However, the landmark set may become less and less representative with increasing collection size and possibly changing music taste. This may have a significant effect on the quality of the projection as, e.g., already observed when applying Landmark MDS for vectorization [28].

2.3 Growing Self-Organizing Map (GSOM)

SOMs are commonly applied for structuring data collections by clustering similar objects into identical or neighboring cells of a two-dimensional grid. In the field of MIR, they have been used in a large number of applications like the *Islands of Music* [22, 24], the *MusicMiner* [20] or *nepTune* [13]. A recent overview on SOM-related MIR publications is given in [30]. Growing Self-Organizing Maps (GSOMs) have the advantage that the structure of the cell grid does not have to be specified prior to training. Instead, they can grow as needed and adapt incrementally to changes in the underlying collection whereas other approaches may always need to generate a new structuring when the grid becomes too small. Growing hierarchical SOMs have been applied in [3, 23, 25]. In contrast, the *BeatlesExplorer* [27] uses a GSOM that grows by adding new cells at the outer boundary. The same approach is used here as a flat structure is desired. For the initialization, a small hexagonal grid of 2×2 cells is chosen. After a regular training phase with a fixed number of iterations, an internal error is computed for each cell as a measure of quality. To this end, the pairwise distance of the objects contained in a cluster is used. Unless the maximum error is below a threshold which is specified as stopping criterion, a new cell is added next to the border cell with the highest error. Here, a very low threshold is used to produce a large map with cells that only contain few songs. In order to reduce visual overlapping of songs in the same cell, song coordinates are computed by taking the distance-weighted mean of the cell's coordinates and those of its direct neighbors. This way, songs are placed slightly off-center.

SOMs generally require the objects they process to be represented as vectors, i.e., elements of a vector space. As the feature representation does not adhere to this condition, some means of vectorization is required. This issue has been discussed in depth in [28] with the recommendation to use MDS for vectorization. For growing collections where only a fraction of the collection may be available for the initial vectorization, the Landmark MDS approach has to be used whereby the songs of the initial collection serve as landmarks. As observed in [28], this may cause a significant drop in the nearest neighbor retrieval precision if many new songs are added – especially if those are very different from the initial ones.

2.4 Stochastic Neighbor Embedding (SNE)

Rather than trying to maintain pairwise distances like MDS, the objective of SNE [9] is to preserve the *probabilities* of points being neighbors. To this end, probability distributions $p_{j|i}$ on how likely it is that the point j is a neighbor of point i are defined based on the input space distances using a Gaussian neighborhood function with width σ_i^2. The algorithm then tries to find suitable coordinates in the output space that lead to (approximately) identical probabilities $q_{j|i}$ for each pair of points. The Kullback-Leibler divergence

$$D_{KL}(p_i, q_i) = \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \tag{1}$$

between the probability distributions $p_{j|i}$ in the input space and $q_{j|i}$ in the output space serves as cost function. Starting from randomly initialized coordinates (close to the origin), a conjugate gradient method is used for finding a (locally) optimal solution.

SNE does not support changes in the collections. However, it is possible to replace the random initialization and instead use the coordinates from the current visualization for those points already present. This way, the search for a new solution (incorporating the new points) starts from the current one. Whilst this does not give any guarantee that the resulting map resembles the old one, chances are that the difference is small.

2.5 Neighbor Retrieval Visualizer (NeRV)

As shown in [32], SNE focuses on the cost of missing similar objects and – by using continuous probabilities to model neighborhood relationships – optimizes a *smoothed* version of the mean recall, $\sum_i D_{KL}(p_i, q_i)$. [1] NeRV additionally takes the *mean smoothed precision* into account by

[1] The *smoothed mean recall* and the *smoothed mean precision* as introduced in [32] are in fact cost functions to be minimized in contrast to

considering also the cost of retrieving dissimilar objects, $\sum_i D_{KL}(q_i, p_i)$. The resulting combined cost function is

$$E = \lambda \sum_i D_{KL}(p_i, q_i) + (1 - \lambda) \sum_i D_{KL}(q_i, p_i) \quad (2)$$

where the parameter $\lambda \in [0, 1]$ controls the trade-off between precision and recall. For $\lambda = 1$, the cost function of SNE is obtained. Another important parameter is the scale of the neighborhoods, σ_i^2, for the computation of the probability distributions $p_{j|i}$ and $q_{j|i}$. Starting with a large value and reducing it stepwise after each optimization step helps to avoid local minima during the conjugate gradient optimization. Like SNE, NeRV does not support collection changes but can be initialized with given coordinates.

3. VISUALIZATION QUALITY MEASUREMENTS

3.1 Test Collection: The Beatles Corpus

For our experiments, we used the well-studied corpus of the 12 official albums of The Beatles containing 180 songs. The albums were added one-by-one to the collection in the order of their release. Distances between the songs was computed as a simple linear combination over three content-based features: Using the CoMIRVA framework [26], we extracted Gaussian Mixture Models of the Mel Frequency Cepstral Coefficients (MFCCs) [19] and the "fluctuation patterns" described in [24]. Furthermore, we derived chord frequency distributions from the publicly available ground truth annotations [8]. This feature combination was chosen to capture common acoustic properties as well as to reduce the problem of hubs [4].

3.2 Quality Measures

3.2.1 *Continuity and Trustworthiness*

A visualization can be considered *trustworthy* if the k nearest neighbors of a point on the display are also neighbors in the original space. Following [12], a respective measure of trustworthiness can be computed by:

$$M_{trustworthiness} = 1 - C(k) \sum_{i=1}^{N} \sum_{j \in U_k(i)} (r_{ij} - k) \quad (3)$$

where r_{ij} is the rank of j in the ordering of the distance from i in the original space, $U_k(i)$ is the set of i's false neighbors in the display, and $C(k) = 2/(Nk(2N - 3k - 1))$ is a constant for obtaining values in $[0, 1]$.

Analogously, the measure of *continuity* considers the k nearest neighbors in the original space and captures how well they are preserved in the visualization [12]:

$$M_{continuity} = 1 - C(k) \sum_{i=1}^{N} \sum_{j \in V_k(i)} (\hat{r}_{ij} - k) \quad (4)$$

Here, $\hat{r}_{ij}$ is the rank of j in the ordering of the distance from i in the visualization and $V_k(i)$ is the set of i's true neighbors missing in the visualized neighborhood.

the traditional definition of precision and recall in information retrieval. For simple "binary" neighborhood definitions, the Kullback-Leibler divergences and the precision-recall measures become equivalent [32].

3.2.2 *Mean Smoothed Rank-Based Precision and Recall*

As a second pair of evaluation measures, it is possible to directly use mean smoothed precision and recall (described in Section 2.5) as error measures. However, these two errors have no upper bound and their scale depends on the dataset. Thus, it would only be possible to compare values for the same sub-collection. As data-independent alternatives, the *mean smoothed rank-based precision and recall* have been proposed in [32]. The idea is to replace the data-dependent distances in the computation of the probability distributions by ranks, i.e., the nearest neighbor gets assigned a distance of 1, the second closest a distance of 2 and so on. The worst case scenario of reversed ranks gives an upper bound that can be used for normalization so that all values lie in $[0, 1]$. [2]

3.2.3 *Mean Position Change*

Finally, a measure was needed to capture the amount of change in the visualization. To this end, we computed the average Euclidean distance between the initial and updated coordinates of all points contained in two consecutive visualizations. In order to obtain normalized values, the point sets were scaled and translated to fit into the unit square. This resulted in a maximal position change of $\sqrt{2}$ for a single point.

3.3 Results

Figure 1 shows the values of the five quality measures for each step of adding another album to the collection. We chose neighborhoods of size $k = 5$ for measurements. The same parameter value was also used for the NeRV and SNE cost functions. The NeRV parameter λ was set to zero to optimize precision as contrast to SNE. Applying the MDS vectorization method on the initial collection, which only contained the first album ("Please Please Me"), yielded 13-dimensional real vectors to be processed by the GSOM. This vectorization was used in all measurements. In comparison to using the full dataset, which would result in 143 dimensions, a small retrieval performance penalty of 1-4% was observed. This matches with the 5% drop in the 10-nearest-neighbor retrieval precision reported in [28] for a 14:166 ratio of landmarks and new songs. The penalty can be expected to increase for more extreme ratios.

Considering only the overall visualization quality, NeRV clearly and consistently yielded the best results. It was surprisingly only slightly outperformed in continuity and recall by SNE, which is specifically optimizing the latter measure while producing significantly worse results for trustworthiness and precision. In general, all methods did similarly well in continuity and recall, i.e., the number of actual neighbors being misplaced in the visualization (false negatives) was rather low. For trustworthiness and precision, the differences were much more obvious indicating

[2] The mean smoothed rank-based measures are still error measures, i.e., a value of zero indicates optimal performance. To obtain consistency with the traditional precision-recall measures, we additionally transform them by $f(x) = 1 - x$ as proposed in [32].

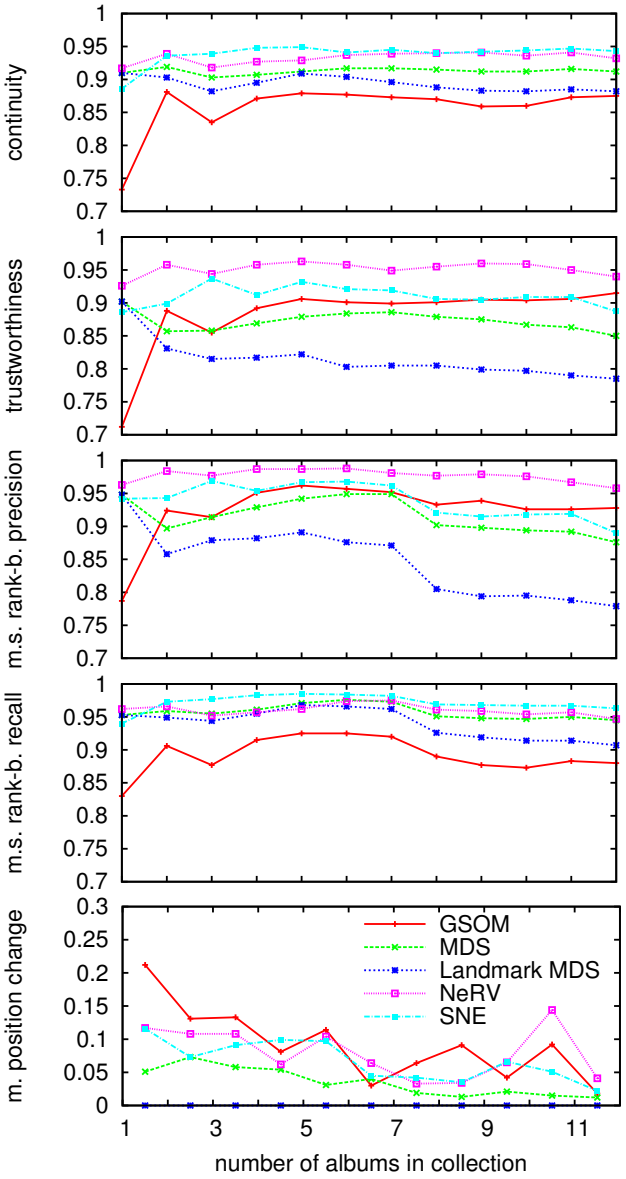

Figure 1. Measurements of the visualization quality and incremental position change. In the top four diagrams, higher values indicate better visualization quality whereas small values of change are desired in the bottom diagram.

issues with neighbors in the visualization that did not belong to the input space neighborhood (false positives). Values were also less constant here. Especially for Landmark MDS, they decreased as more and more songs were added. A significant drop in precision could be observed when the eighth album ("Sgt. Pepper's Lonely Hearts Club Band") was added to the collection.

Whilst NeRV generated an excellent visualization in each step, the difference between them was often much higher than for the other algorithms. Only the GSOM performed worse here. This might at least partly be due to the GSOM's grid-based structure, which leads to a kind of position quantization. Hence, when a song is re-assigned to a neighboring cell (the smallest possible position change), its position already changes roughly by the grid cell width. Landmark MDS was the clear winner here with no position changes at all, but this came at the cost of degrading visualization quality. MDS appears to offer a good compromise: The amount of change was lower than for the other methods (except Landmark MDS) and it visibly decreased as the fraction of new songs became smaller. At the same time, the recall and continuity value were comparable to NeRV. Only trustworthiness and precision were about 10% below NeRV.

4. USER STUDY

In this section, we describe the design and results of a comparative user study as qualitative evaluation. In order to reduce the effort for the participants, we only tested MDS, GSOM, and NeRV. Landmark MDS was not considered because there is no observable change for existing songs that could be evaluated and SNE because of its close relation to the superior NeRV method. The following research questions were addressed: *How well can users follow the changes when adding a new album, what algorithm do users prefer and why?* The study was conducted using the Beatles corpus and the same visualizations evaluated in the quantitative measurements described in Section 3.

4.1 Study Design

The user study was designed as follows: In a pre-interview, we gathered the users' demographic information, computer skills and their knowledge about The Beatles and map-based visualizations. Then, a lab experiment with within-participants design was performed, i.e., each participant viewed all three visualizations. We used a *Latin square design* to reduce the bias caused by the order in which participants were introduced to the different layout algorithms, i.e., a third of the participants tested MDS first, GSOM second and NeRV at last, whereas another third evaluated the algorithms in the order GSOM-NeRV-MDS and another third used the ordering NeRV-MDS-GSOM. After a short trial period of free interactions, we asked all participants to perform a memorization task in the form of a game similar to thimblerig. At each step, starting with the first album of The Beatles, three songs were highlighted randomly for five seconds as shown in Figure 2. Shortly after, participants were asked to track these songs during the transition of the visualization to the next step. The task was to find the previously highlighted songs in the updated layout. There was no time limit. The number of errors made by each user was recorded as a measurement for their performance. Thus, for each algorithm, we gathered errors made at each of the 11 (as we used 12 albums) steps. After the test on each algorithm, users were asked how well they were able to track the changes in the songs' layout. Finally, we asked the participants what algorithm they preferred. The whole procedure took about 30 minutes.

4.2 Study Results

We performed the user study with 19 subjects. They were between 23 and 38 years old (28 on average), eight women and eleven men. 68% of them were professional computer

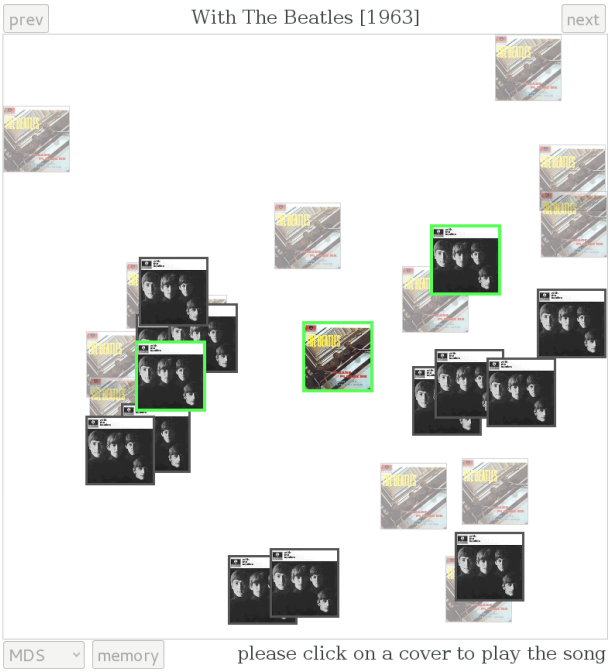

Figure 2. Graphical user interface for the study. [4] Here, a collection containing the first two albums of The Beatles is visualized using the MDS algorithm. The participants were asked to track all three highlighted songs (covers with green border) as they move during a transition.

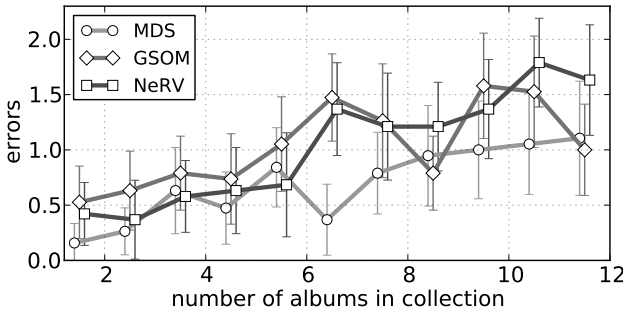

Figure 3. Mean of memorization errors for each transition, and confidence intervals ($\alpha = 0.05$).

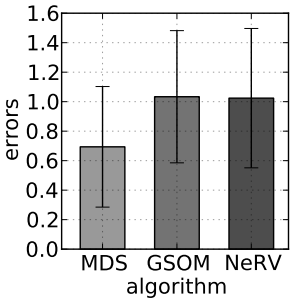

Figure 4. Mean memorization errors over all transitions, and confidence intervals ($\alpha = 0.05$).

users, whereas the rest had intermediate level. 68% of participants were students or PhD students in computer science. Only 21% were well acquainted with the music of The Beatles, 47% were familiar with map visualization.

Thirteen participants chose MDS, six chose NeRV and one chose GSOM as their favorite layout algorithm (one chose both MDS and NeRV). Users described MDS to result in less positional changes, NeRV to better preserve cluster structures and GSOM to have less overlappings. The results of the memorization task are shown in Figure 3. With an increasing number of albums, it became more difficult to find the requested songs. At later steps, participants noted that it was hard to select the desired songs because of many overlappings. Also, at some transitions, e.g., from six to seven albums, there are significant differences between algorithms. In total, the layout generated by the MDS algorithm resulted into the least mistakes as shown in Figure 4. Unfortunately, our results are not statistically significant, presumably due to the low number of participants. However, there was a strong tendency to the MDS algorithm. Most participants (14) achieved their best results using the MDS algorithm.

5. CONCLUSIONS

In this paper, we evaluated five popular methods for producing two-dimensional maps of music collections with respect to their ability to deal with incrementally growing collections. Two of these, GSOMs and Landmark MDS, have this ability by design. The other three, MDS, SNE, and NeRV, had to be slightly modified for this purpose. We conducted objective measurements – tracking the visualization quality and the change of positions over time – and a user study. Considering both the measured results and the feedback from the participants of the user study, MDS can be proclaimed as "winner". This is much to our surprise. We did not expect MDS (in combination with the Procrustes analysis for alignment) would produce such nice incremental visualization updates as observed here. Unlike the second best, NeRV, MDS has furthermore the advantage that it does not incorporate randomness and that it is guaranteed to find a *globally* optimal solution.

It is possible though highly unlikely that the outcome was in part caused by the choice of the collection. Therefore, we will conduct further experiments to verify the results using different datasets and also different media like texts and images. Another highly promising possibility for future work is to modify NeRV to better support incremental collection changes – e.g., by introducing an additional weighted term to its cost function (Equation 2) that penalizes position changes. This way, a new method could be designed that results in less dramatic position changes but maintains NeRV's superior performance for trustworthiness and precision. For reproducibility and to encourage possible testing with further methods, the dataset (in particular the distance matrix used as input for the algorithms) will be made publicly available.

[4] For better readability, the screenshot was taken using a much smaller screen resolution than in the user study. An online demo is available at http://demos.dke-research.de/beatles-history-explorer/

Acknowledgments We would like to thank all participants of the user study and J. Venna et al. for sharing their NeRV implementation code. The work in this paper has been funded in part by the German Federal Ministry of Education and Science (BMBF) through the Forschungscampus STIMULATE under Grant No. FKZ:03FO16103A.

6. REFERENCES

[1] P. Cano, M. Kaltenbrunner, F. Gouyon, and E. Batlle. On the use of fastmap for audio retrieval and browsing. In *ISMIR*, 2002.

[2] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Adv. in Neural Information Processing Systems 15*, 2002.

[3] M. Dopler, M. Schedl, T. Pohle, and P. Knees. Accessing music collections via representative cluster prototypes in a hierarchical organization scheme. In *ISMIR*, 2008.

[4] A. Flexer, D. Schnitzer, M. Gasser, and T. Pohle. Combining features reduces hubness in audio similarity. In *ISMIR*, 2010.

[5] M. Gasser and A. Flexer. Fm4 soundpark: Audio-based music recommendation in everyday use. In *Proc. of 6th Sound and Music Computing Conf.*, 2009.

[6] D. Gleich, M. Rasmussen, L. Zhukov, and K. Lang. The World of Music: SDP layout of high dimensional data. In *Info Vis*, 2005.

[7] J.C. Gower and G.B. Dijksterhuis. *Procrustes Problems*. Oxford University Press, 2004.

[8] C. Harte, M.B. Sandler, S.A. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *ISMIR*, 2005.

[9] G.E. Hinton and S.T. Roweis. Stochastic neighbor embedding. In *Adv. in Neural Information Processing Systems 15*, 2002.

[10] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2002.

[11] C. F. Julia and S. Jorda. SongExplorer: a tabletop application for exploring large collections of songs. In *ISMIR*, 2009.

[12] S. Kaski, J. Nikkila, M. Oja, J. Venna, P. Toronen, and E. Castren. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, 4(1):48, 2003.

[13] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Exploring Music Collections in Virtual Landscapes. *IEEE MultiMedia*, 14(3):46–54, 2007.

[14] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[15] J.B. Kruskal and M. Wish. *Multidimensional Scaling*. Sage, 1986.

[16] A.S. Lillie. Musicbox: Navigating the space of your music. Master's thesis, MIT, 2008.

[17] S. Lloyd. Automatic playlist generation and music library visualisation with timbral similarity measures. Master's thesis, Queen Mary Univ. of London, 2009.

[18] D. Lübbers and M. Jarke. Adaptive multimodal exploration of music collections. In *ISMIR*, 2009.

[19] M. Mandel and D. Ellis. Song-level features and support vector machines for music classification. In *ISMIR*, 2005.

[20] F. Mörchen, A. Ultsch, M. Nöcker, and C. Stamm. Databionic visualization of music collections according to perceptual distance. In *ISMIR*, 2005.

[21] R. Neumayer, M. Dittenbach, and A. Rauber. PlaySOM and PocketSOMPlayer, alternative interfaces to large music collections. In *ISMIR*, 2005.

[22] E. Pampalk, S. Dixon, and G. Widmer. Exploring music collections by browsing different views. In *ISMIR*, 2003.

[23] E. Pampalk, A. Flexer, and G. Widmer. Hierarchical organization and description of music collections at the artist level. In *Proc. of 9th Eu. Conf. on Research and Advanced Technology for Digital Libraries*, 2005.

[24] E. Pampalk, A. Rauber, and D. Merkl. Content-based organization and visualization of music archives. In *Proc. of 10th ACM int. Conf. on Multimedia*, 2002.

[25] A. Rauber, E. Pampalk, and D. Merkl. Using psychoacoustic models and self-organizing maps to create a hierarchical structuring of music by musical styles. In *ISMIR*, 2002.

[26] M. Schedl. The CoMIRVA Toolkit for Visualizing Music-Related Data. Technical0 report, Johannes Kepler University Linz, 2006.

[27] S. Stober and A. Nürnberger. Towards user-adaptive structuring and organization of music collections. In *Proc. of 6th int. workshop on Adaptive Multimedia Retrieval*, 2008.

[28] S. Stober and A. Nürnberger. Analyzing the impact of data vectorization on distance relations. In *2011 IEEE Int. Conf. on Multimedia and Expo*, 2011.

[29] S. Stober and A. Nürnberger. Musicgalaxy: A multi-focus zoomable interface for multi-facet exploration of music collections. In *Exploring Music Contents*, volume 6684 of *LNCS*, pages 273–302, 2011.

[30] S. Stober and A. Nürnberger. Adaptive music retrieval – a state of the art. *Multimedia Tools and Applications*, 65(3):467–494, 2013.

[31] R. van Gulik and F. Vignoli. Visual playlist generation on the artist map. In *ISMIR*, 2005.

[32] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research*, 11:451–490, 2010.

[33] C.K.I. Williams. On a connection between kernel pca and metric multidimensional scaling. *Machine Learning*, 46(1-3):11–19, 2002.