A General Approach for Similarity-based Linear Projections Using a Genetic Algorithm

James A. Mouradian, Bernd Hamann and René Rosenbaum

Institute for Data Analysis and Visualization (IDAV), Department of Computer Science, University of California, Davis, Davis, CA 95616-8562, U.S.A.

ABSTRACT

A widely applicable approach to visualizing properties of high-dimensional data is to view the data as a linear projection into two- or three-dimensional space. However, developing an appropriate linear projection is often difficult. Information can be lost during the projection process, and many linear projection methods only apply to a narrow range of qualities the data may exhibit. We propose a general-purpose genetic algorithm to develop linear projections of high-dimensional data sets which preserve a specified quality of the data set as much as possible. The obtained results show that the algorithm converges quickly and reliably for a variety of different data sets.

Keywords: High-dimensional data, linear projections, genetic algorithms, data visualization

1. INTRODUCTION

Many application domains produce data of far more than three dimensions. Due to the limitations of human cognition, visually representing such data is an ongoing challenge. Many different approaches have already been developed, but most are designed for low-dimensional data and are of limited use for data of higher dimensionality. A reasonable approach to deal with truly high-dimensional data are projections. The development of effective and intuitively understandable projections of high-dimensional data, however, remains a visualization challenge. The general strategy of projections is to transform data from high-dimensional space into a two- or three-dimensional display space. Due to the fact that this significantly reduces the number of dimensions of the data, information loss is the main problem associated with this technique. Projection methods are well-studied and have proven useful in a variety of application domains. They are usually designed to preserve pre-defined aspects of the data, such as variance of a high-dimensional point cloud data set to keep information loss small. These aspects can be quickly calculated by incrementally constraining the search space, leading to low complexity in finding an appropriate solution. A typical example is the well-known principle component analysis (PCA). However, such projection methods are highly specialized. More general features about the data, such as similarity in the relative point distances, are lost. Projections preserving such features cannot be simply determined as their computation is usually NP-hard.

In this paper, we propose a general-purpose projection method for the determination of an appropriate projection for nearly arbitrary similarity metrics using a genetic algorithm (GA). The algorithm probabilistically produces linear projections which attempt to accurately preserve the chosen metric of a given data set. By evaluating potential projection matrices for the data, the algorithm repeatedly refines estimates of what an optimal similarity-preserving projection might be. The algorithm terminates when it reaches a stability threshold, at which point candidate projection matrices do not appear to better preserve the similarity in question than their predecessors. The algorithm is not dependent on a specific similarity and can be applied to a broad variety of metrics. The results obtained for artificial and real-world data sets demonstrate that this approach is fast and reliable. It is able to produce projections that preserve the desired characteristics of the data, including similarities which require high complexity to assess.

Further author information: (Send correspondence to James Mouradian)
James Mouradian: E-mail: jamouradian@ucdavis.edu
Bernd Hamann: E-mail: hamann@cs.ucdavis.edu
René Rosenbaum: E-mail: rosenbaum@ieee.org

In Section 2, we briefly survey research related to our method, and delineate our method from existing strategies. Section 3 is concerned with the introduction of the main strategy of our method and potential similarity metrics. The results obtained from applying the proposed strategy to a variety of different data sets are shown in Section 4. Section 5 is dedicated to show options for an appropriate parameterization of the algorithm. We conclude in Section 6 that by the use of our approach, projections preserving general features of a high-dimensional data set can be determined quickly and with high accuracy.

2. RELATED RESEARCH

One way to visualize high-dimensional point cloud data is to project the data from the high-dimensional space into low-dimensional space, usually two- or three-dimensional display space. Such a dimension reduction can be viewed as a data smoothing, simplification, or compression operation, since information is lost and existing structure in the data may not be preserved. Thus, projections must be carefully chosen.

Research has been conducted in a variety of fields to address this problem. Solutions can be classified into two main approaches: *linear* and *non-linear projections*.

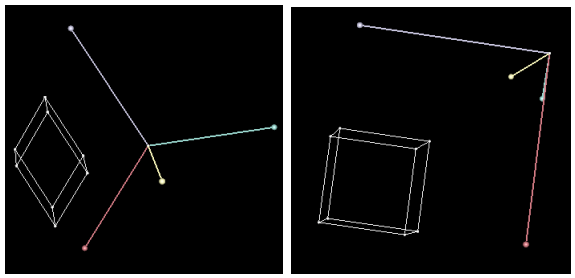
Linear projections, also often referred to as geometric transformations, apply a static transformation matrix to all the data points. For projections into two-dimensional space, a projection plane is positioned in the data space. Several methods have been studied, such as principal[1] and independent component analysis,[2] linear discriminant analysis and the related Fisher's linear discriminant, or multidimensional scaling,[3] among others.[4] Even methods resulting in more than one projection have been proposed.[5] The methods are well-understood, robust and easy to implement.

An appropriate position and orientation of the projection plane depends on the respective objectives of the visualization and the projection approach. The low complexity of the mentioned strategies stems from incrementally narrowing the search space by excluding dimensions or parts of the data that are of less interest for the respective visualization goal. A typical example is the widely applied PCA and its numerous extensions[6,7] which project correlated dimensions into less correlated dimensions based on the variance in individual dimensions. Dimension reduction is achieved by eliminating the components with low variance. Variance, as well as the new space, are determined by the eigenvectors of the covariance matrix, which can be quickly calculated.

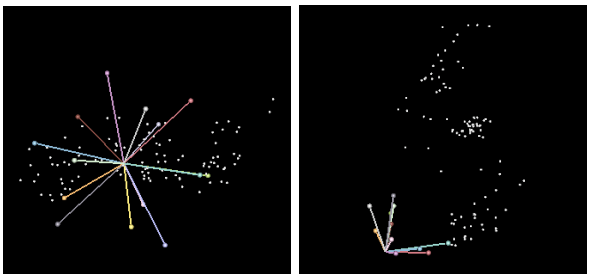
Although many linear projection methods are appropriate when the visualization goals are known in advance, the produced projections are often too specific to provide a global, unbiased overview of the data. Less specific views are often useful in exploratory visualization where data properties may be unknown, especially before data analysis. The calculation of projections which preserve arbitrary features or provide general overviews of the data, however, can be extremely complex and time-consuming. The approach introduced in this paper determines such projections in a very short amount of time. Due to its modularity, it can also be used as a part of a larger framework, such as.[8] In order to cope with obscurity of projected data and to provide a solution for multiple view systems,[9] the introduced method supports many different similarity metrics.

Non-linear projections, such as Sammons plots,[10] self-organizing maps (SOM),[11] and non-linear multidimensional scaling (MDS),[12] are not limited by a static transformation matrix and thus are more flexible for data presentation. However, such projections might unrecognizably deform the geometry of the original data and thus are not guaranteed to reliably show its genuine properties. The Sammons plot attempts to preserve the distance between data points in both the original and projection spaces, and therefore has a similar objective to parts of the introduced method. However, its calculation is difficult and time-consuming, and its individual data point projections are difficult to interpret because they are non-linear. Due to our focus on linear projections, we do not present a comparison with non-linear projection methods.

Genetic algorithms (GA), typically viewed as a part of evolutionary computation, are mostly used to handle and provide stochastic solutions for NP-hard problems (e.g.,[13]). Their eligibility for dimension reduction has already been shown for applications spreading from data base research[14] and pattern classification,[15] to domain-specific problems in the sciences.[16] Recent research also indicates that they can be applied to visualization, such as the calculation of SOMs[17] and the support of specific data features, such as clusters,[18] as well as training projection algorithms.[19] To our knowledge, no research has been published toward using GAs to deal with the complexity in calculating of projections for nearly arbitrary similarity metrics as proposed in this paper.



(a) Two projections of a three-dimensional cube stored in four-dimensional space



(b) Two projections of a fourteen-dimensional data set

Figure 1. **Comparing linear projections:** In both sub-figure (a) and sub-figure (b), a high-dimensional data set is projected into three-dimensional space in two ways. In each pair, the left image is a linear projection with basis vectors distributed as equiangularly as possible in attempt to preserve qualities of the data space itself, while the right image is a data-dependent projection which attempts to preserve the pairwise distances between data points. In the data-dependent projections, the cube in sub-figure (a) is not distorted, and three clusters become apparent in sub-figure (b).

3. DETERMINING APPROPRIATE PROJECTIONS USING A GENETIC ALGORITHM

We introduce a fast, general-purpose GA in order to linearly project n-dimensional data into an m-dimensional projection space ($m \ll n$) while best preserving a single measurable quality of the data. After introducing *the main strategy* of our approach, we provide different *potential similarity metrics* that can serve to assess projection quality.

3.1 The Main Strategy

Our proposed strategy to find an appropriate projection is to incrementally refine the accuracy of an approximated projection using a GA. Projections which most accurately preserve the selected quality of the data are used as input to generate new projections in attempt to converge toward a global optimum. We measure quality using the projected data points, as the data points and not the data space itself should be meaningfully projected to better gain insight about the data (see Figure 1).

Many different similarity metrics can be used (see Section 3.3), and the data can be projected from and into an arbitrary number of dimensions. For simplicity of example, but without loss of generality to the algorithm, we try to preserve pairwise Euclidean distances between data points in high-dimensional space, and project the data into three-dimensional space for the remainder of this section.

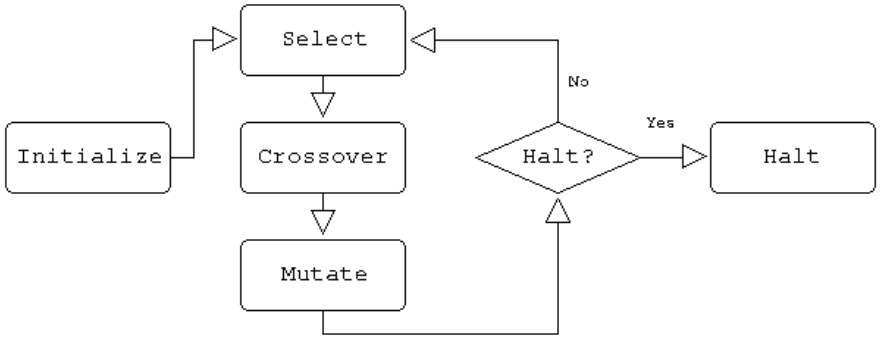


Figure 2. **Genetic algorithm flowchart:** *Initialize* generates a large number of individual solution estimates (individuals). Incremental refinement is achieved by *Selecting* the best individuals, performing *Crossover* of the selected individuals to produce new ones, and *Mutating* these new individuals to explore the search space. An appropriate projection is considered to be found when the refinement process no longer yields significant improvement.

A potential solution, or *individual*, is a set of n three-dimensional basis vectors, where n is the number of dimensions in the data set. We specify individuals in spherical coordinates using the three attributes, (R, θ, ϕ).

3.2 The Individual Steps

The outline of the algorithm and the individual stages are illustrated in Figure 2.

Initialize The initialization step of the algorithm randomly generates the initial *population* or *generation* of individuals to uniformly sample the search space of solutions, and makes any initial computations that future stages of the algorithm may rely on. The individuals in the presented approach are sets of n three-dimensional basis vectors, each consisting of a randomly chosen R, Θ, and Φ. In order to have a reference value against which to compare prospective projections, the relative distances of all point pairs in high-dimensional space are computed and stored. Let these quantities make up the n-dimensional quality $Q_{n-space}$.

Select The *selection* stage of the algorithm determines which individuals are the best, or most *fit*, of the current population, in order to guide the production of future generations of individuals in attempt to converge toward the global optimum. In the presented approach, for each candidate projection, the relative distances of all point pairs in the projected three-dimensional space, $Q_{3-space}$, are calculated. Individuals who have a high *fitness*, or whose $Q_{3-space}$ values closely resemble $Q_{n-space}$, are chosen as the *selection* used to generate individuals for the next iteration of the algorithm.

Crossover During the crossover step, the most fit individuals of the current generation are combined with one another repeatedly to form the generation for the next iteration of the algorithm. Combining individuals based on fitness alone without consideration for their other properties permits early exploration of the solution space rather than early exploitation of a few possible solutions. In our approach, largely different projection matrices which evaluate to similar fitness values can be combined, resulting in radically new projection matrices which have not been considered. This reduces the probability of the algorithm "getting stuck" in local optima. We propose to pseudorandomly pick pairs of parents from the *selection* to each generate two offspring according to one of the two schemes illustrated in Figure 3. Any number of crossover methods are possible, but strategies taking into account the specifics of linear projections have potential to yield to better results.

Mutate After each individual is generated during the crossover step, it undergoes random *mutation*, where attributes of each individual are randomly altered. Favorable mutations increase the fitness of individuals over each iteration, probabilistically leading to an overall near-optimum projection. The combination of selection and mutation is largely responsible for the algorithm's convergence toward a global optimum.

Mutation can be performed by randomly flipping individuals' bits, or by randomly adjusting realized attributes of individuals. For our application, mutating individuals by flipping random bits could be detrimental to the algorithm's performance. This is due to the fact that our individuals consist of a series of a very small range of values: $0 < R \leq 1$, $0 \leq \theta \leq \pi$, and $0 \leq \phi \leq 2\pi$. Randomly flipping bits in floating-point numbers is likely to generate values which are out of these narrow bounds, and additional computation steps would be required to either normalize or discard and regenerate meaningless values. Moreover, by having the ability to

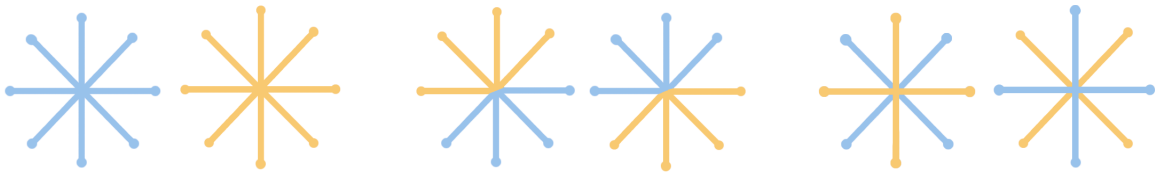


Figure 3. **Crossover** is the process by which new individuals are created. A pair of individuals, i.e., sets of basis vectors (left), can be split into contiguous halves and recombined (middle), or alternating halves and recombined (right) to form two new offspring.

reduce the magnitude of mutations over time, we allow individuals to remain optimal during later stages of the algorithm, rather than spontaneously leaping toward sub-optimal configurations.

Our proposed mutation method is as follows: Randomly select an attribute A from R, θ, ϕ from each basis vector according to a probability of mutation P_{mut}. The attribute A with maximal magnitude A_{max} is adjusted by a random value $Rand_m$, bounded by a maximum magnitude, Mut_{max}, as follows:

$$A_{next} \leftarrow A_{previous} + Rand_m,\ -(\text{Mut}_{max} \times A_{max}) \leq Rand_m \leq +(\text{Mut}_{max} \times A_{max}).$$

We decrease the P_{mut} and Mut_{max} geometrically every few iterations to allow the algorithm to refine its adjustments to individuals over time.

Halt After generating sufficiently fit individuals, the algorithm terminates, and reports its most fit individual. For threshold problems, the algorithm terminates when an individual reaches a threshold fitness; for problems with known optima, the algorithm can terminate within some tolerance of a known optimum. However, the problem we concern ourselves with is neither a threshold problem, nor a problem where the fitness value of a global optimum projection can always be known in advance. As such, we set the algorithm to terminate when its progress in developing fit projections has reached the point of diminishing returns, or when further iterations of the algorithm do not appear to significantly further increase the fitness of individuals.

We define our termination condition as follows: Let I be the number of recent iterations to consider, J be the total number of iterations, F_k be the average increase of the fitness of the fittest individual in the last k iterations, and C be a non-negative constant less than one. The algorithm terminates when $F_I \leq CF_J$.

3.3 Potential Similarity Metrics

The proposed method is modular and widely independent of the metric used to determine the quality of the projection. Thus, it has a broad range of possible application domains. The only requirement for a potential quality metric is that it must be comparable in order to judge whether a configuration provides better quality than another. A certain quality metric can be introduced by adapting only the "select" stage of the algorithm.

Examples for similarity metrics that can be employed by our methods include common approaches, such as the Euclidian distance,[20,21] the dot product between the points' high-dimensional position vectors,[22] Mahalanobis,[23] Minkowski,[21] and Manhattan distance[21] as well as novel approaches, such as the one proposed by.[24] A good source for the selection of an meaningful metric for data exploration is given by.[25]

Our method supports global as well as detail metrics. Global metrics lead to a single, "global" quantitative value that is used for comparison, e.g., the accumulated relative distances between pairs of points in our space. For instance, a global Euclidean distance metric could be described as $\sum 1 - (\frac{d_{i,j}}{\Delta_{i,j}})^2$ for each pair of points i and j, where $d_{i,j}$ is the distance between the points in three-dimensional space, and $\Delta_{i,j}$ is the distance between the points in n-dimensional space. An optimal projection minimizes this sum. A global dot product metric might minimize $\sum 1 - (\frac{\psi_i \bullet \psi_j}{x_i \bullet x_j})^2$ where ψ_k is a point's position vector in n-dimensional space, and x_k is a point's position vector in three-dimensional space.

Detail metrics consider individual point-to-point similarities. They result in multiple quantitative values that can be used for comparison, such as by the maximal dissimilarity or by the percentage of points with strong similarities. For instance, a detail Euclidean distance metric might be the standard deviation of the quantities $\frac{\Delta_{i,j}}{d_{i,j}}$, where $d_{i,j}$ is the distance between the points in three-dimensional space and $\Delta_{i,j}$ is the distance between the points in n-dimensional space. An optimum projection minimizes this standard deviation. Such a projection may incur a greater accumulated sum measured using a global metric, but the distance between each individual pair of points is projected more proportionately to other pairs of points in the projection.

4. RESULTS

We experimented with and tested our introduced method with a variety of data sets, parameters, and termination conditions. As GAs are highly dependent on the *used parameters*, meaningful values we obtained from our experiments are summarized below. We present visual results we obtained from "*proof-of-concept*" data sets showing the plausibility of the method and *real-world data* sets underpinning its usefulness for a broad variety of potential application domains.

We implemented and tested three similarity metrics: the global and detail Euclidean distance metrics and the global dot product metric discussed in Section 3.3.

4.1 Used Parameters

After experimenting with various parameters, we found that the algorithm converged quickly with largely reproducible results. Across all the data sets covered in this section we applied the following parameter values:

$< Initial\ Population >$	**1000**
$< Selection\ Size >$	**40**
$< Generation\ Size >$	**200**
$< P_{mut} >$	begins at **0.20** and decreases geometrically by a factor of 0.9 every 15 iterations
$< \mathrm{Mut}_{max} >$	begins at **0.20** and decreases geometrically by a factor of 0.9 every 15 iterations
$< I >$	**25**
$< C >$	**0.01**

4.2 Proof of Concept

Genetic algorithms unfortunately offer no guarantee of convergence or correctness. We therefore evaluated the algorithm using some basic test data to ensure its plausibility before applying it to real-world data sets.

Basic Spatial Accuracy In order to check for visual plausibility, we projected the corners of a three-dimensional cube given in four-dimensional space into three-dimensional space (see Figure 1). An optimal projection of the four-dimensional data space would equiangularly distribute the four basis vectors in three-dimensional space, distorting the cube, while the algorithm produces a more accurate representation of the cube using both the global and local Euclidean distance metrics.

Preserving Qualities across Similar Data Sets We also used two artificial data sets of similar structure but different dimensionality to evaluate that the algorithm would produce similar projections for both.

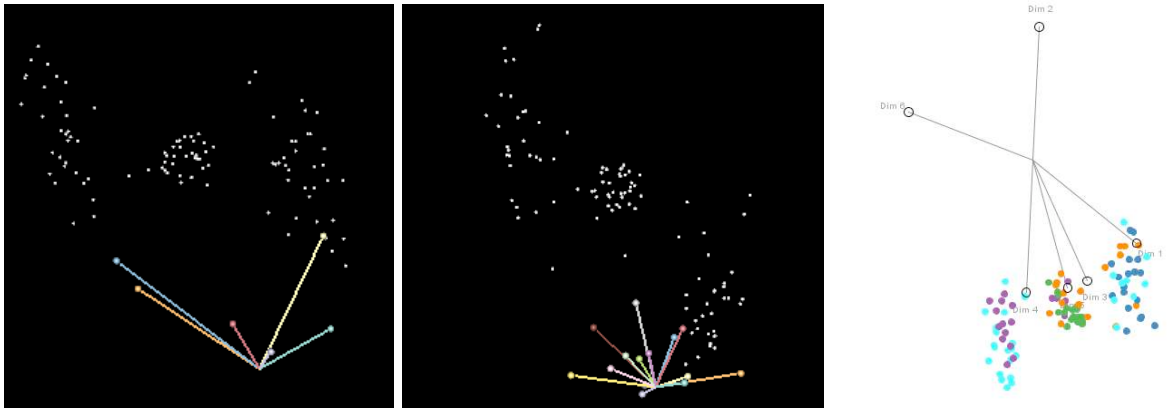


Figure 4. **Similar projections:** A 6- (left) and 14-dimensional data set (center) projected by our algorithm with the global Euclidean distance metric in three space. As shown for the 6-dimensional data set, the revealed structure in the data is comparable to the outcome of a high-accuracy brute-force algorithm (right). For these data sets, all three of our metrics yielded vastly similar projections.

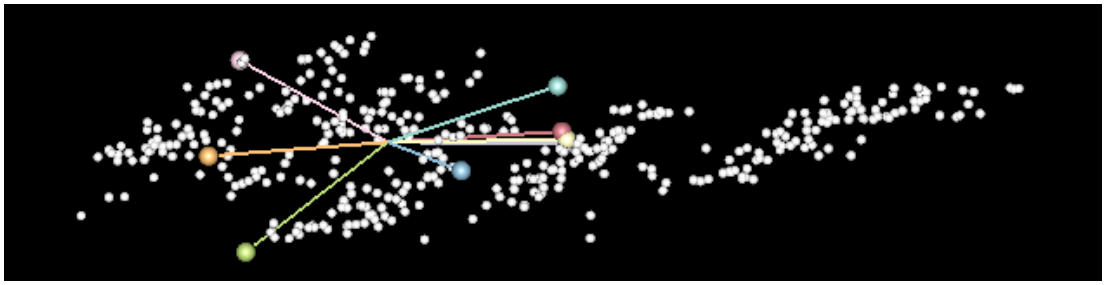


Figure 5. **The "cars" data set** projected using the global Euclidean distance metric: The projection determined by the proposed method reveals clusters in the data, visible as striations across the main body of data points.

Each of the data sets is known to have three clusters and various linear correlations. The projections of each data set exhibit similar structures with regard to the global topology and local structure of the individual clusters (see Figure 4, left and center).

Consistence with Another Approach Projections obtained by the application of our approach with the global Euclidean distance metric were also compared to a high-resolution, brute-force strategy which exhaustively compares various projections to find a near-global-optimum solution. The visual results shown in Figure 4, left and right, are comparable and mainly differ in orientation and the scaling. This leads to the conclusion that our approach also found a near-optimum solution for the considered data set.

Consistence with Various Metrics Similarity metrics which resemble one another should at least sometimes produce projections which resemble one another. We were able to demonstrate that both the global and local Euclidean distance metrics provided similar projections across several test data sets, as well as several real-world data sets. The global dot product metric produced comparable results, but leads to a different final projection due to the focus on another quality within the data (see Figures 5 and 6). Performing this verification ensured that each of the different error metrics was plausible after verifying the correctness of the global Euclidean distance metric.

4.3 Real-world data

We applied the algorithm to the well-known cars data set,[26] studied extensively by.[27] It contains 398 instances and exhibits various clusters in the ten-dimensional data space as well as in its lower-dimensional sub-spaces. We project 391 instances of eight dimensions of the data. The introduced method clearly separates three out of five clusters in data space (see Figure 5) using the Euclidean distance metric; other clusters are not as clearly separated, but a grouping can still be noticed. The use of the dot product metric leads to a much better representation, emphasizing the importance of using multiple similarity metrics (see Figure 6).

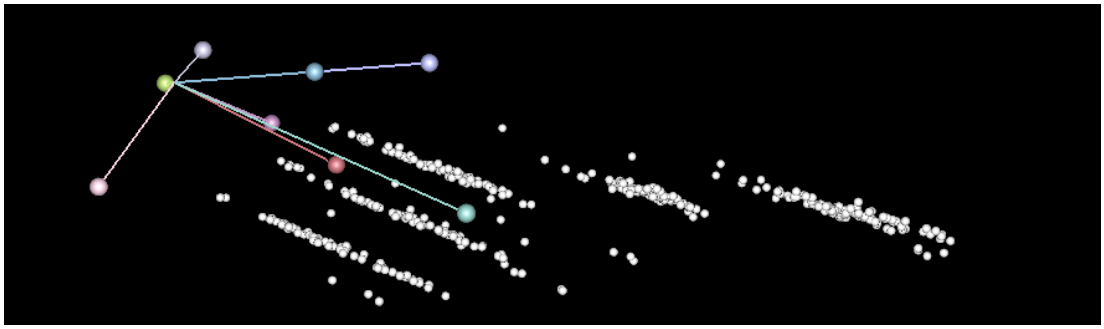


Figure 6. **The "cars" data set** projected using the global dot product metric: The projection determined using the global dot product metric greatly pronounces the five distinct clusters of the data set and also highlights the outliers of the individual clusters.

We also tested the algorithm on a large scale data set made available by our collaborators from the Air Quality Research Center (AQRC) at UC Davis. The data represent the size and chemical composition of air particles assessed by mass spectrometric techniques. The data is high-dimensional (255 dimensions) and consists of 210,000 individual particles. It was collected from two distinct sample sites, whereby data from one site was obtained in two different sampling campaigns. Our approach, as well as some existing strategies, failed to calculate meaningful projections due to the high number of data points. In order to reduce the data volume we sampled the data uniformly to one thousand data points, and considering the thirteen most important of the dimensions, selected with PCA, were able to calculate a meaningful projection in roughly twenty minutes. The two main clusters are highly apparent, and are associated with the two sampling sites from which the data was taken. The stretched cluster on the left reveals the inhomogeneous character of the data resulting from the repeated sampling of only one of the sample sites. The extremely repetitive computations performed by the algorithm make it infeasible to use the algorithm on the original data set in its entirey; however, sampling techniques appears viable, as similar findings have also been published for this data set.[28]

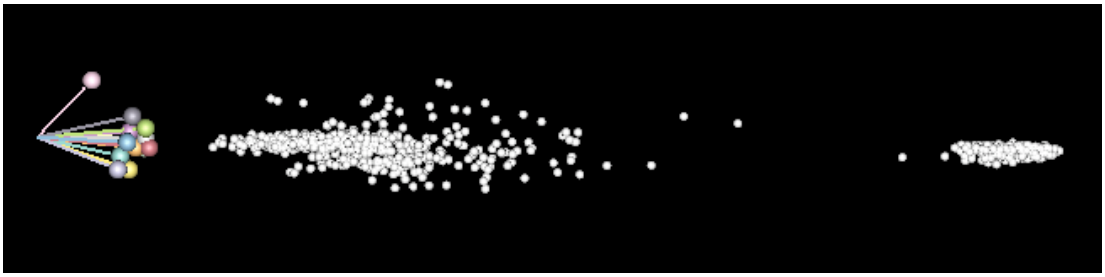


Figure 7. **Air quality data:** The introduced strategy for the incremental determination of a projection leads to the expected two clusters corresponding to the two different sample sites.

4.4 Convergence: Efficiency and Accuracy

The time complexity of each iteration of the algorithm is linear in the generation size, and dependent on the similarity metric supplied as well as the data set provided. For instance, in the case of the Euclidean distance metric, a naive algorithm is linear in the number of dimension: $O(d)$ to compute the distance between two points in $d-$space, and quadratic in the number of data points ($O(n^2)$) to examine all $\binom{n}{2}$ pairs of points in the data set. A user can achieve increased performance by storing pre-computed data and using optimized formulae.

When compared to the brute-force approach mentioned in Section 4.2, the method we present is orders of magnitude faster. The presented method is able to produce projections of six-dimensional, one-thousand-point data sets in approximately eleven minutes, while the mentioned brute-force method requires approximately forty-six hours to process a comparable volume of data. However, due to the algorithm's quadratic complexity and many repeated computations during incremental refining of the projection, its scalability, especially with regard to the number of data points given, remains limited.

The parameters introduced in Section 3 and discussed in the following section influence the running time and accuracy of the algorithm. We provide figures focusing on performance data of the algorithm as a function of one or more highly influential parameters in order to provide a general overview of trends of the algorithm's behavior.

Number of Points	Number of Dimensions	Approximate Algorithm Run Time
100	14	30 Seconds
391	8	1 Minute
1000	6	11 Minutes
1000	13	20 Minutes

Figure 8. **Run time:** Approximate algorithm run times for the data sets discussed in Sections 4.2 and 4.3. Each estimate was developed by averaging a minimum of three runs on a 2-GHz CPU with the parameters used in Section 4.1.

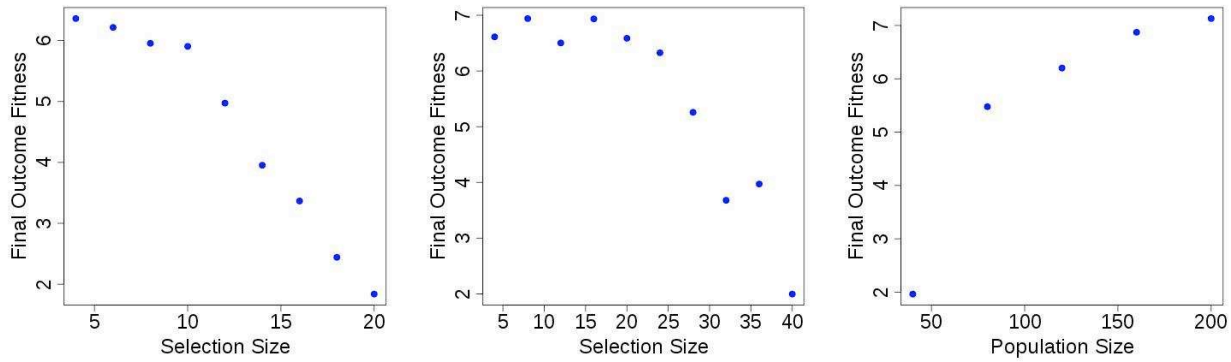


Figure 9. **Selection and Population Sizes:** Increasing selection size too greatly decreases final outcome fitness against a population size of 20 (left). Increasing selection size too greatly decreases final outcome fitness against a population size of 40 (center). Increasing population size significantly has an impact on the final outcome of the algorithm (right).

5. PARAMETER VARIATION

Varying the algorithm's parameters usually have a tremendous influence on GAs. In our case, it strongly influences convergence. In this section we present a general overview of how various parameters can influence the algorithm in different stages. We briefly discuss the most important parameters introduced in Section 3.2, as each has definite impact on the algorithm's outcome.

Unless otherwise stated, the following data was collected with parameter values as stated in Section 4.1. Each data point was taken as an average of 10 executions of the algorithm applied to a proof-of-concept data set consisting of 100 data points in 14 dimensions.

5.1 Number of Individuals: Population and Selection

Population size and selection size must be sufficiently large for the algorithm to produce appropriate results. When the selection size is too close to the population size, the algorithm does not converge toward near-optimum solutions. However, when the ratio is appropriate, a relatively small selection and population can be used to find near-optimal projections. Large initial populations do not appear to affect the algorithm's convergence.

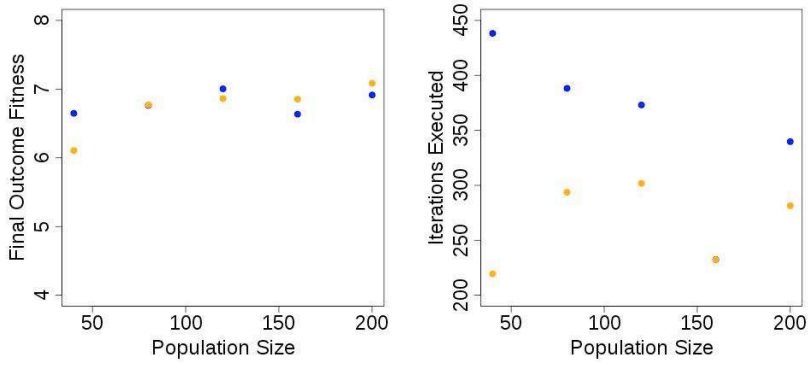


Figure 10. **Crossover:** the final outcome fitness of individuals produced by the algorithm without crossover (blue) are quite similar to those produced by the algorithm with the interleaved crossover (orange) method (left). The algorithm may execute many fewer iterations when using the interleaved crossover method as opposed to when copying individuals from the prior generation directly (right). Blue data points not visible are eclipsed by orange data points. For these results, 20 iterations were considered for the termination condition.

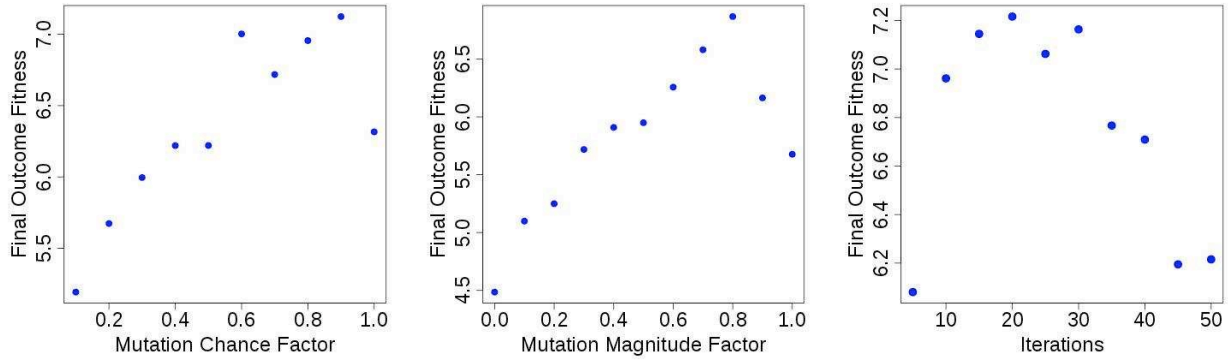


Figure 11. **Mutation:** Geometrically decreasing the chance of mutation decreases the overall outcome of the algorithm (left). Geometrically decreasing the maximum magnitude of mutation too greatly is detrimental to the overall outcome of the algorithm (center). As long as the mutation chance decreases a few times through the course of the algorithm, the overall outcome of the algorithm improves (right).

5.2 Crossover

The effects of utilizing crossover rather than copying individuals from the prior generation directly are subtle. For large population and selection sizes, crossover may greatly reduce the number of iterations the algorithm executes before terminating without decreasing the final outcome individual of the algorithm (See: Figure 10). Neither of the crossover methods introduced in 3.2 appeared to strongly impact the final outcome of the algorithm under these conditions. For small population and selection sizes, crossover appears to positively impact the average fitness of the final selected individual produced by the algorithm; however, the average fitness of the final selected individual given such small population and selection sizes may be sub-optimal. It is worth noting that for some data sets, the space of possible linear projections may not be complex enough to necessitate crossover to find a global optimal projection. This especially applies data sets with only a few local optima.

5.3 Mutation

We vary the attributes R, Θ, Φ of basis vectors in three-dimensional space according to a specified probability, within some maximum magnitude which decreases over time. Insufficient mutation chance and magnitude are detrimental to the algorithm's convergence.

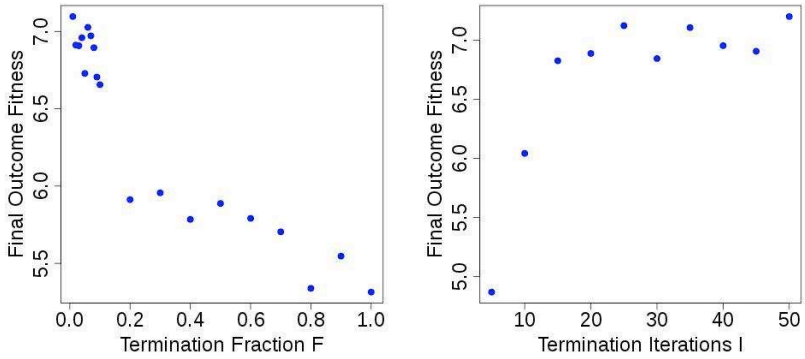


Figure 12. **Termination:** Requiring a small (0.01-0.10) F_I has a positive impact on final outcome fitness (left). Increasing F_I too greatly decreases overall outcome fitness. Considering an insufficient number of iterations in the termination condition decreases overall outcome fitness (right).

5.4 Termination Conditions

The algorithm terminates when the most recent I iterations fail to keep up some minimum fraction C of the average increase in fitness of the most fit individual across all iterations. With decreased C, the algorithm plateaus further before terminating. With increased I, the algorithm is required to continue iterating into a plateau, potentially even to an unnecessary extent. When C becomes sufficiently large, I will force the algorithm to run until it has reached its minimum number of iterations, after which point it will stop.

6. CONCLUSIONS

The presented genetic algorithm, which determines appropriate linear projections to preserve arbitrarily selected qualities of point cloud data given in high-dimensional space, converges effectively and reliably across a variety of data sets. The results obtained using three different underlying quality metrics are plausible, and with the two presented global metrics, are confirmed against a brute-force approach to produce near-optimal projections. Due to the algorithm's modularity, a user can supply the algorithm with a broad variety of similarity metrics with minimal effort, making the algorithm viable for users who are performing initial exploratory visualization of a data set that can be effectively sampled. The run-time behavior of our method is magnitudes lower than related brute-force strategies; our presented approach is able to process six-dimensional, one-thousand-point data sets in approximately eleven minutes, as opposed to the forty-six hours required by a related brute-force approach. However, appropriate parameter value choices must be made in order to obtain high-quality results. The properties of the introduced approach make it a very useful method to show specific trends and qualities in high-dimensional data which are difficult to present with existing projection and visualization techniques.

Future research is open in several areas. The fitness evaluation function is responsible for a large portion of the algorithm's run time and is embarrassingly parallel. Since fitness evaluation is the same process in each individual, running the algorithm on parallel GPU architectures should decrease runtime significantly. Further investigation regarding the significance of crossover may lead to the ability to reliably decrease the population and selection sizes used in the algorithm, as well as the algorithm's run time. We will also perform research directed at visualizing contextual information about the structural decomposition of the data by using clustering techniques. Such contextual information can serve to disambiguate or clarify the relationships between points in high-dimensional space, despite loss incurred by low-dimensional projections, and increase the scalability of the algorithm to larger data sets.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of Deutsche Forschungsgemeinschaft (DFG) for partially funding this research (#RO3755/1-1). The authors also acknowledge Oliver Kreylos for providing the Vrui framework as well as Jeremy Bottleson for developing the InfoVisualizer software both used within our research.

REFERENCES

[1] Jolliffe, I. T., [*Principal component analysis*], Springer, New York (2002).
[2] Hyvärinen, A., Karhunen, J., and Oja, E., [*Independent Component Analysis*], Wiley, New York (2001).
[3] Cox, T. F. and Cox, M., [*Multidimensional Scaling, Second Edition*], Chapman and Hall/CRC, 2 ed. (2000).
[4] Fodor, I., "A survey of dimension reduction techniques," *LLNL technical report* (June 2002).
[5] Friedman, J. and Tukey, J., "A projection pursuit algorithm for exploratory data analysis," *Computers, IEEE Transactions on* **C-23**(9), 881–890 (1974).
[6] Koren, Y. and Carmel, L., "Robust linear dimensionality reduction," *IEEE Transactions on Visualization and Computer Graphics* **10**(4), 459–470 (2004).
[7] Dhillon, I. S., Modha, D. S., and Spangler, W. S., "Visualizing class structure of multidimensional data," *Proceedings of the 30th symposium on the interface: Computing science and statistics* **30**, 488–493 (1998).
[8] Jeong, D. H., Ziemkiewicz, C., Fisher, B., Ribarsky, W., and Chang, R., "iPCA: an interactive system for PCA-based visual analytics," *Computer Graphics Forum* **28**(3), 767–774 (2009).

[9] Friedman, J. H., "Exploratory projection pursuit," *Journal of the American Statistical Association* **82**, 249–266 (Mar. 1987).

[10] Sammon, J. W., "A nonlinear mapping for data structure analysis," *IEEE Transactions on Computers* **18**(5), 401–409 (1969).

[11] Kohonen, T., "The self-organizing map," *Proceedings of the IEEE* **78**(9), 1464–1480 (1990).

[12] Venna, J. and Kaski, S., "Local multidimensional scaling," *Neural Networks* **19**, 889–899 (July 2006).

[13] Colorni, A., Dorigo, M., and Maniezzo, V., "Genetic algorithms and highly constrained problems: The Time-Table case," (1990).

[14] Golmah, V. and Parvizian, J., "Visualization and the understanding of multidimensional data using genetic algorithms : Case study of load patterns of electricity customers," *International Journal of Database Theory and Application* **3**, 41–56 (Dec. 2010).

[15] Panicker, R. C. and Puthusserypady, S., "A constrained genetic algorithm for efficient dimensionality reduction for pattern classification," in [*Computational Intelligence and Security, International Conference on*], **0**, 424–427, IEEE Computer Society, Los Alamitos, CA, USA (2007).

[16] Guo, Q., Wu, W., Questier, F., Massart, D. L., Boucon, C., and de Jong, S., "Sequential projection pursuit using genetic algorithms for data mining of analytical data," *Analytical Chemistry* **72**, 2846–2855 (July 2000).

[17] Romero, G., Guervós, J. J. M., Valdivieso, P. A. C., Castellano, J. G., and Arenas, M. G., "Genetic algorithm visualization using self-organizing maps," in [*Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*], *PPSN VII*, 442–451, Springer-Verlag, London, UK, UK (2002).

[18] McCaffrey, J. D., "An empirical study of categorical dataset visualization using a simulated bee colony clustering algorithm," in [*Proceedings of the 5th International Symposium on Advances in Visual Computing: Part I*], *ISVC '09*, 179–188, Springer-Verlag, Berlin, Heidelberg (2009).

[19] Raymer, M., Punch, W., Goodman, E., Kuhn, L., and Jain, A., "Dimensionality reduction using genetic algorithms," *Evolutionary Computation, IEEE Transactions on* **4**(2), 164–171 (2000).

[20] Ogras, Ümit Y. and Ferhatosmanoglu, H., "Dimensionality reduction using magnitude and shape approximations," in [*Proceedings of the twelfth international conference on Information and knowledge management*], *CIKM '03*, 99–107, ACM, New York, NY, USA (2003).

[21] Bagherjeiran, A. and Eick, C. F., "Distance function learning for supervised similarity assessment," in [*Case-Based Reasoning on Images and Signals*], Perner, P., ed., **73**, 91–126, Springer Berlin Heidelberg, Berlin, Heidelberg (2008).

[22] Kim, D., Lee, K., Lee, D., and Lee, K. H., "A kernel-based subtractive clustering method," *Pattern Recognition Letters* **26**, 879–891 (May 2005).

[23] Poranne, R., Gotsman, C., and Keren, D., "3D surface reconstruction using a generalized distance function," *Computer Graphics Forum* **29**, 2479–2491 (Dec. 2010).

[24] Tejada, E., Minghim, R., and Nonato, L. G., "On improved projection techniques to support visual exploration of multi-dimensional data sets," *Information Visualization* **2**, 218–231 (Dec. 2003).

[25] Aggarwal, C. C., "Towards systematic design of distance functions for data mining applications," in [*Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*], *KDD '03*, 9–18, ACM, New York, NY, USA (2003).

[26] UCI repository of machine learning databases, "Autos mpg," (1983).

[27] Kandogan, E., "Visualizing multi-dimensional clusters, trends, and outliers using star coordinates," in [*Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*], *KDD '01*, 107–116, ACM, New York, NY, USA (2001).

[28] Engel, D., Rosenbaum, R., Hamann, B., and Hagen, H., "Structural decomposition trees," *Computer Graphics Forum, EuroVis* **30**, 921–930 (June 2011).