The LloydRelaxer An Approach to Minimize Scaling Effects for Multivariate Projections

Dirk J. Lehmann and Holger Theisel

Abstract—Star Coordinates are a popular projection technique in order to analyze and to disclose characteristic patterns of multidimensional data. Unfortunately, the shape, appearance, and distribution of such patterns are strongly affected by the given scaling of the data and can mislead the projection-based data analysis. In an extreme case, patterns might be more related to the choice of scaling than to the data themselves. Thus, we present the *LloydRelaxer*: a tool to minimize scaling-based effects in Star Coordinates. Our algorithm enforces a scaling configuration for which the data explains the observed patterns better than any scaling of them could do. It does so by an iterative minimizing and optimization process based on Voronoi diagrams and on the Lloyd relaxation within the projection space. We evaluate and test our approach by real benchmark multidimensional data of the UCI data repository.

Index Terms—Information Visualization, Multidimensional Data, Scaling, Star Coordinates, Projections

1 Introduction

Finding good projections of high-dimensional data sets is one of the standard problems in Information Visualization. In many applications, the input data can be interpreted as points in a high-dimensional data space; its visualization requires a suitable projection onto the 2D screen space. To find good projections, a variety of automatic, interactive, and hybrid techniques have been proposed in the literature. Especially the concept of Star Coordinates [21], [22], [24] became increasingly popular. Star Coordinates define an affine projection from the nD data space onto a 2D projection space. They visualize the dataset completely with respect to its n dimensions and m data records. Nevertheless, the analysis of multidimensional data still remains a challenging task, denoted by the term *curse of dimensionality*. This means that the analysts have to consider a set of effects and aspects, such as over-plotting [31], scalability [23], [26], distortions [24], relevance [1], [4], loss of visual contrast [29], [38] and so forth, in advance of or during a successful analysis process. These issues have already been addressed in many contexts within our community. Even though they are far from completely being solved, they are well considered and a lot of research goes in this direction. On the other side, one additional important aspect that strongly influences the analysis has not been taken notice of: the *choice of scaling*.

To illustrate the problem addressed in this paper, let us consider a thought experiment: imagine a (toy) climate data set consisting of m measurements of temperature, air pressure, rainfall, and humidity. This means that every measurement can be considered as a point in a 4D data space. All four dimensions are physically unrelated in the sense that they are measured in different (and unrelated) physical units. This means that the ratio of the values in different dimensions is arbitrary and has no meaning. Changing the ratio by scaling one dimension generally changes the result of the projection, raising the following question: if a projection shows an interesting structure, is it really a property of the data or is it the result of an unfavorable scaling of one or more dimensions? We search for projections that are independent of the scaling of one dimension. For our example, it should not make a difference if, e.g., the temperature is measured in *Celsius*, *Fahrenheit*, or *Kelvin*. The simplest solution for this is normalization: every dimension is normalized independently such that the data values are in the range of $[0, 1]$ or $[-1, 1]$, or they are translated and scaled such that the standard deviation is 1 around the mean/median of each dimension (=whitening). While these approaches are common, they may falsify the results.

How can we find such patterns that minimize scaling effects? Our idea is to look for a scaling that maximizes the expressiveness of global patterns, i.e., structures that represent the complete dataset. Thus, a global pattern could only be seen and explained by an interplay between all (or most of) dimensions of the nD data. Since a badly chosen scaling might cause that local patterns are more weighted and thus more prominent in the multivariate projection – which might mislead the user's data interpretation process – we are interested in finding the scaling configuration which stresses the global patterns best. Thus, we propose an approach based on computational geometry: For an arbitrary Star Coordinate projection, we are looking for a scaling $\mathbf{k}$ of all coordinate axes such that the Voronoi diagram of the projected points becomes most regular and close to its centroidal Voronoi diagram. For this, each projected point needs to be shifted to its centroid position of its related Voronoi cell. This gives an iterative process – based on Fortune's algorithm [14] – which converges to the centroidal Voronoi diagram, known as Lloyd relaxation [30].

In order to get the properties of the data involved, we adopt this Lloyd relaxation in a way that in each iteration our approach is looking for a scaling $\mathbf{k}$, which minimizes the difference between the regular Lloyd relaxing step and the projected point set, denoted as *LloydRelaxer* of the data. The final pattern of the projection can be explained best by the data instead of the choice of scaling. Thus, our *LloydRelaxer* minimizes scaling effects for Star Coordinate

Dirk J. Lehmann is with Rey Juan Carlos University of Madrid, Spain and University of Magdeburg, Germany; dirk@isg.cs.uni-magdeburg.de
Holger Theisel is with University of Magdeburg; theisel@ovgu.de

projections. Please note that our approach focuses on finding a good scaling but not on finding a good axis orientation. The axis orientation is user input here. Finding a good axis orientation is orthogonal to the issues addressed in this work and already discussed in a set of works, such as [16], [24], [25], [26], [28]. Moreover, to the best of our knowledge, we present the first approach addressing the scaling issue for projections, and consequently, there are no trivial solutions we could compare our approach with. In fact, there are no feasible alternatives. Furthermore, there is no ground truth available we can compare our approach with. This applies due to the fact that an appropriate scaling is not measurable for the time being. Thus, we consider the scaling issue from a heuristic point of view in order to tackle it (cf. Sec. 6). Having this in mind, Figure 1 (right) shows the resulting projections and related final Voronoi diagram configurations that minimize the scaling effects for the dataset 1 and 2.

To summarize, in this paper we focus on linear projections that can be represented as Star Coordinates. We search for projections that are independent of the scaling of each dimension. Our approach is that for a given data set and a given linear projection we explore the space of all scalings of all dimensions to find optimal scalings such that the projections show the most informative results. For this, we will first show that scaling the dimensions in data space is equivalent to scaling the coordinate axes in Star Coordinates. Second, we define what an informative projection is. For this, we follow [13], who solved the problem for the special case of 2D data, i.e., scatterplots: We prefer projections with an as-regular-as-possible point layout to make sure that the shown structures are inherent to the data and not to the scaling of one dimension. Third, we describe a numerical optimization to find the best scaling of each dimension as an iterative process. We analyze the algorithm and apply it to several test data sets. In conclusion, the contributions of this paper are:

- we explain and discuss the scaling issue in visualization,
- we introduce an iterative approach called *LloydRelaxer* which minimizes scaling artefacts for affine and global multivariate projections, i.e., Star Coordinates, w.r.t. an initial linear/affine scaling in the data,
- we introduce an implementation scheme for an interactive *LloydRelaxer*-based descaling tool, and
- we present empirical experiments with the *LloydRelaxer* based on synthetica and real nD datasets.

Subsequently, we discuss work that is related to ours.

2 Related Work

Since we combine and extend a set of techniques, our related work comes from both the field of projection techniques and the field of computational geometry.

Projection Techniques: The affine multivariate projection technique named Star Coordinates was established in 2000 by Kandogan [21], [22], extended by Lehmann and Theisel in 2013 to an orthographic preserving version [24], and in 2016 generalized [25] to a universal projection concept that unifies projective, affine, and orthographic projections from an *n*D data space onto a 2D projection space. Star Coordinates became very popular in recent years for analyzing multidimensional data, and their properties are well investigated both in a technical sense [36] as well as regarding their perception properties [12]. Thus, for our scaling-minimizing concept we rely on them, too.

Due to the fact that the space of projections contains an infinite number of elements, a couple of techniques were described to purposefully select interesting projections or at least to navigate through this space in an effective way. An interpolation scheme between a pre-selected relevant and initial set of projections was introduced in [33], based on considering the projections as corner points of a convex polygon and defining the interpolated projections within by the use of mean value coordinates. How such an initial set of relevant sets of projections might be found, was presented in [26]. There, a small but complete set with $n/2$ projections results from the idea of avoiding projections which can be mapped onto each other by the use of affine transformations. In [39], an approach is introduced in order to interactively probe projections aiming to support the understanding of certain dimension arrangements. Even though such approaches might help to find multivariate projections that show interesting data patterns, none of these have considered scaling effects of the data at all. Thus, it remains unclear for the user whether the provided patterns are caused by data or by the scaling. Consequently, we provide projection-based techniques to justify whether these patterns are caused by the data or by the scaling and we present interaction schemes to navigate through the projection space while showing only projections where scaling effects are minimized. We do so by using schemes from computational geometry.

Computational Geometry: The two concepts of the Delaunay triangulation and the Voronoi diagram are standard approaches in the field of computational geometry. Both approaches give a starting point for complex geometrical operations on 2D point sets (cf. Figure 2 (top - left)), such as minimum spanning trees [35] or Dijkstra's shortest path [7] and many more. Considering a 2D point set, the Delaunay triangulation [6], [8], [17], [18] is a regular triangulation where the circumcircles of any tessellation's triangle contain an empty point set (cf. Figure 2 (top - middle)). The dual problem to this is the Voronoi diagram [3] of this 2D point set (cf. Figure 2 (top - right)). The Voronoi diagram (VD) equates to the segmentation of the point set into a set of adjacent convex polygons, denoted as Voronoi cells, where each polygon contains exactly one 2D point of this set and where each edge (except borders) is equidistant w.r.t. a certain metric to their two closest adjacent 2D points. Typically, the formally mentioned p2-norm or Euclidean distance is used as metric for the purpose of constructing a Voronoi diagram. We do so throughout this work. Larger point densities are qualitatively expressed by more but smaller Voronoi cells and vice versa. So the VD is directly related to the distribution of the point set [11]. In 1987, Steven Fortune [14], [15] presented a sweepline algorithm to calculate the Voronoi diagram of a 2D point set with a number of m points in $O(m \log m)$. Fortune's algorithm is still the fastest for usual point configurations. Hence, we also utilize it to generate Voronoi diagrams for this work.

In addition, each Voronoi cell – and thus each 2D point – can be assigned by a 2D centroid, being the center of gravity by assuming a homogeneous mass distribution in that cell. If the 2D point and its 2D centroid are equal for all those points of the set, the Voronoi diagram is named centroidal Voronoi diagram (CVD) [10], [20] (cf. Figure 2 (bottom - right)). This equates to a perfect uniform distribution of the points in that set. In order to iteratively transform an arbitrary VD into a CVD, Stuart Lloyd provided a popular algorithm, known as the *Lloyd Relaxation* [9], [30], [37]. Briefly spoken, the 2D points are iteratively shifted to the positions of their related 2D centroids until this process finally converges in a CVD (cf. Figure 2 (middle up to bottom)). Our approach is based on this Lloyd Relaxation. In the following, we review how such scaling issues are currently handled.

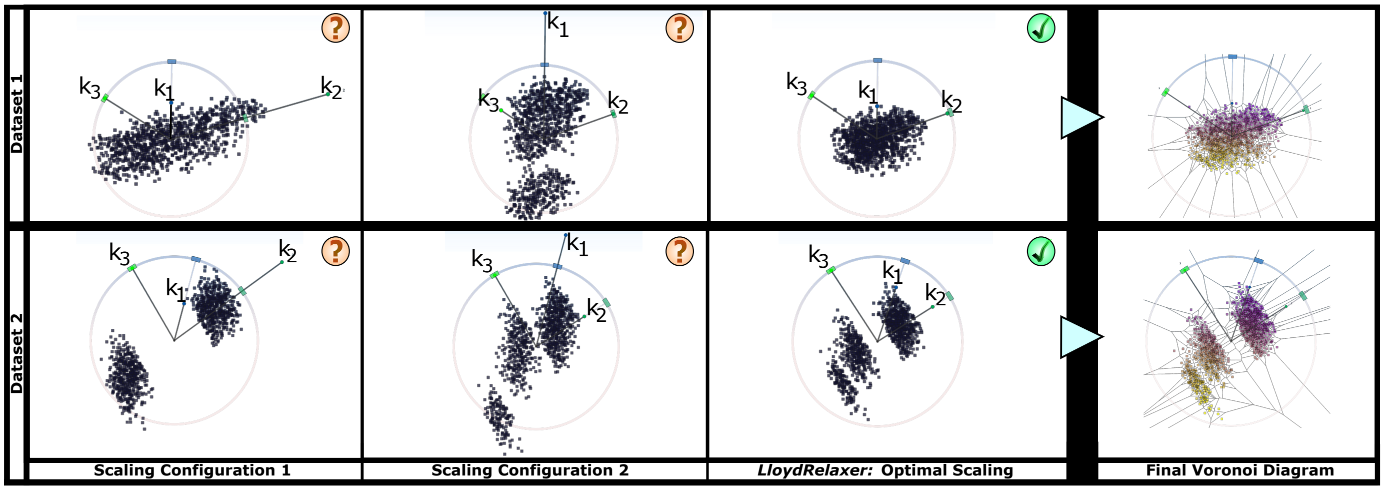

Fig. 1. *LloydRelaxer* in order to minimize scaling effects: (from left) The first two sub-figures per row show projections for different scaling configurations of two different datasets (dataset 1 on top, dataset 2 on bottom) with $m = 1000$ data records and $n = 3$ dimensions each: The scaling strongly influences which patterns result, even though they should be solely dependent on the data. Within the last two sub-figures per row, our *LloydRelaxer* minimizes such scaling effects. This way, the most data-related patterns occur. In detail: (top) The scaling configuration 1 intends a (weak) multi-linear correlation between the three dimensions. On the contrary, the scaling configuration 2 intends two clusters. Which observation can explain the data best? Our optimal scaling discloses that both observations are false and cannot be explained by the data. Instead, the data might not contain a clear pattern at all. (bottom) The scaling configuration 1 of the data intends two clusters. In contrast, the different scaling configuration 2 of the data suggests three clusters. Finally, our optimal scaling confirms the second observation that the data contains (at least) three clusters, which explains the data best.

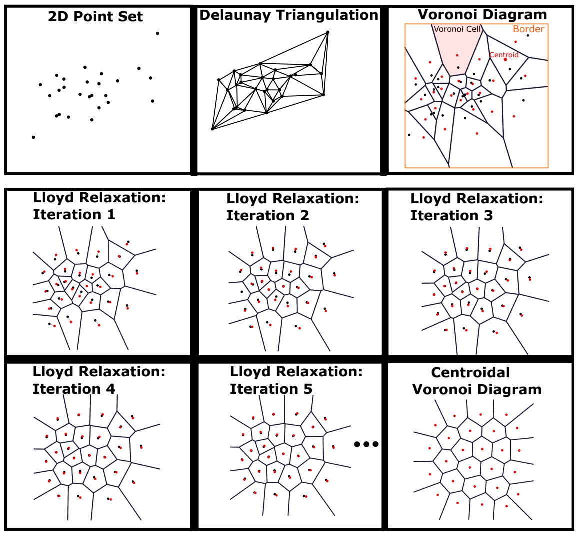
2D Point Set
Delaunay Triangulation
Voronoi Diagram
Voronoi Cell
Border
Centroid
Lloyd Relaxation: Iteration 1
Lloyd Relaxation: Iteration 2
Lloyd Relaxation: Iteration 3
Lloyd Relaxation: Iteration 4
Lloyd Relaxation: Iteration 5
Centroidal Voronoi Diagram

Fig. 2. Schemes for the Delaunay Triangulation, Voronoi Diagram, Lloyd Relaxation, and Centroidal Voronoi Diagram for a 2D point set.

Scaling vs. Projections: The issue of finding an appropriate scaling for the data for visualization purposes is currently reduced to the issue of finding an appropriate aspect ratio for bivariate scatterplots. Regarding this, Fink et al. [13] propose to select a convenient aspect ratio for those plots, meaning to find a value s in a way that the scaling of this plot with s for x- and $(1/s)$ for the y-coordinates reveals a good view to the user. They argue that a good aspect ratio is given if the related Delaunay triangulation of the set of 2D plot points has certain geometrical properties, such as, e.g., a large minimal angle or a small total edge length. Hence, finding the one degree of freedom s is given (and solved) as an optimization issue over the mentioned triangulation. Talbot et al. [41] go in a similar direction, but differ in their criteria to select the aspect ratio, which is here considered best if the arc length for certain representatives in the plot is minimized. There are many similar approaches, please see further references in [13]. Both approaches generate convincing bivariate scatterplots, i.e., plots which might look comfortable to the user (and to us, too). Nevertheless, it remains unclear if these bivariate plots show patterns of the data or just scaling artifacts as described in Section 1. Furthermore, to the best of our knowledge, there are currently no further techniques available focusing on the optimal choice of scaling for bivariate or multivariate projections.

3 What is an Informative Scaling?

The scaling issue occurs for almost every scaling, due to the fact that a scaling of nD data is usually influenced by the culture (kilometer or miles, degree Celsius or degree Fahrenheit, liters or gallons etc.) or the community/job (milliliter or liter, mile or lightyear, nanosecond or year). This is interesting, since it means that our culture influences which structures appear in a dataset (note that we are not talking about perception or cognition). Thus, for the time being, there is no choice of scaling that is better than another; no ordering relation over scalings is known, but we know that scaling might negatively distort the structures. That is a dilemma: each scaling might be equally good or poor with respect to the underlying domain. Even though we would know that our initial scaling misleads the "meaningful" structures, we cannot ask for a better one until we can define what "better" means: how can we find a quantitative data model in order to distinguish a "good" scaling, which gives us an informative view into the domain, from a "bad" view, which just reflects scaling artifacts? Without having a scaling ground truth on which our models can be evaluated, it is challenging to find such a model.

However, all this is interesting and opens a wide research area, but it is far beyond the focus of our paper. Thus, in order to become able to address the scaling issue and to overcome this dilemma at all, we set ourselves to a heuristic point of view, meaning that we define a hypothesis what a good model for scaling is, in order to confirm this model later empirically (or even to falsify it).

For our scaling model, we follow and generalize [13]: Now, we define a most informative view as this scaling $\mathbf{K}$ that regularizes the Voronoi diagram of the projected points best. Here, the term "regular" means to minimize the variance of the Voronoi cell sizes under varying $\mathbf{K}$.

This minimization process involves (and thus interlocks) each dimension equivalently. Thus, from a geometrical point of view, we expect that the global patterns of the data unfold, appear, and get visible (almost) independent of the initial data scaling. In detail, structures that can be explained by the interplay between a set of dimensions are preferred, while rather local structures – e.g., structures that appear just by playing with the scaling of a single dimension – are suppressed. This way, with our most informative view we get the most global structure enhancing view. Our empirical results confirm this assumption.

To prepare further discussion, we subsequently introduce the used technical notations and the description of the scaling issue from a technical perspective.

4 Notations and Background

In this work, we label mathematical objects as follows:

- a (real) scalar value is given by lower case letter s,
- a (real) column vector is given as bold lower case letter $\mathbf{s}$,
- a real matrix is given as bold upper case letter $\mathbf{S}$.

We avoid to mention the dimensionality of a vector or matrix if it is clear from the context. In addition, we label the unit matrix as $\mathbf{I}$, the one vector as $\mathbf{1}$, and the zero vector as $\mathbf{0}$.

We define the Hadamard product for two $n \times m$ matrices as element-wise multiplication

$$\mathbf{C} = \mathbf{A} \circ \mathbf{B} \quad \leftrightarrow \quad c_{i,j} = a_{i,j} \cdot b_{i,j}. \tag{1}$$

For one $n \times m$ matrix and one $n \times 1$ vector, the Hadamard product is given by

$$\mathbf{C} = \mathbf{A} \circ \mathbf{b} \quad \leftrightarrow \quad c_{i,j} = a_{i,j} \cdot b_i. \tag{2}$$

We define n as the dimensionality of the data space, and m as the number of data points. Then, the j-th point in the data space can be written as $\mathbf{d}_j = (d_{1,j}, ..., d_{n,j})^T$, and the matrix of all data points is the $n \times m$ matrix

$$\mathbf{D} = (\mathbf{d}_1, ..., \mathbf{d}_m). \tag{3}$$

Star Coordinates are defined by setting a 2D coordinate axis $\mathbf{a}_i = (x_i\ y_i)^T$ for every dimension. Such coordinate axes $\mathbf{a}_i$ are usually denoted as *anchor points* of the Star Coordinate projection. The projection is then defined by the $2 \times n$ projection matrix

$$\mathbf{A} = (\mathbf{a}_1, ..., \mathbf{a}_n). \tag{4}$$

An alternative representation of matrix $\mathbf{A}$ is in polar form

$$\mathbf{A} = \overline{\mathbf{A}} \circ \mathbf{k}, \tag{5}$$

being a partition into the normalized matrix $\overline{\mathbf{A}}$ and its scaling $\mathbf{k}$,

$$\overline{\mathbf{A}} = (\bar{\mathbf{a}}_1, ..., \bar{\mathbf{a}}_n), \tag{6}$$

$$\mathbf{k} = (k_1, ..., k_n)^T \tag{7}$$

and

$$k_i = ||\mathbf{a}_i|| \quad , \quad \bar{\mathbf{a}}_i = \mathbf{a}_i / k_i. \tag{8}$$

The direction points $\bar{\mathbf{a}}_i$ encode the direction in unit length (by reflecting the angle of the direction) and the scalars k_i encode the scaling of the projection in Euclidean length w.r.t. anchor points $\mathbf{a}_i = \bar{\mathbf{a}}_i \cdot k_i$. This gives an intuitive approach to visualize the projection matrix $\mathbf{A}$: considering the 2D origin $\mathbf{o}$ in image space, the direction points are visualized as box on a unit circle at $(\bar{\mathbf{a}}_i - \mathbf{o})$, the anchor points as (tiny) circles given at $(\mathbf{a}_i - \mathbf{o})$, and the scaling k_i equates to the length of the line $(\mathbf{a}_i - \mathbf{o})$. The visualization scheme is illustrated in Figure 3.
This way, the 2D projection $\mathbf{p}_j$ of the data point $\mathbf{d}_j$ is $\mathbf{p}_j = \mathbf{A} \cdot \mathbf{d}_j$, and the matrix $\mathbf{P}$ of all projected data points is the $2 \times m$ matrix

$$\mathbf{P} = \mathbf{A} \cdot \mathbf{D} = (\overline{\mathbf{A}} \circ \mathbf{k}) \cdot \mathbf{D}. \tag{9}$$

Since this work is based on the observation that the properties of the data $\mathbf{d}_j$ are influenced by scaling operations, we subsequently give a detailed introduction to this scaling issue.

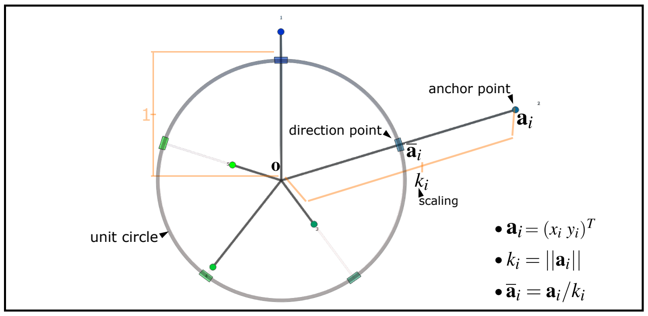
1
anchor point
$\mathbf{a}_i$
direction point
$\bar{\mathbf{a}}_i$
k_i
scaling
o
unit circle
• $\mathbf{a}_i = (x_i\ y_i)^T$
• $k_i = ||\mathbf{a}_i||$
• $\bar{\mathbf{a}}_i = \mathbf{a}_i / k_i$

Fig. 3. Visualization scheme for a projection matrix $\mathbf{A}$ for case $n = 5$.

5 The Scaling Issue

The scaling issue describes that the scaling, i.e., a linear transformation of the data, strongly influences the geometrical properties of the data. Thus, the results of a data analysis are influenced by the scaling and might reflect the underlying scaling much better than the underlying data, which is illustrated in the following.

5.1 Theory of Scaling

Giving an nD data record $\mathbf{d} = (d_1, ..., d_n)^T$ to standard basis $\mathbf{I}$ as

$$\mathbf{d} = \mathbf{d} \cdot \mathbf{I}. \tag{10}$$

The scaling $\mathbf{k} = (k_1, ..., k_n)^T$ for the data record $\mathbf{d}$ is then given by

$$\mathbf{d}' = \mathbf{d} \circ \mathbf{k} = \mathbf{d} \cdot (\mathbf{I} \circ \mathbf{k}), \tag{11}$$

which equates to a basis transformation of the data record $\mathbf{d}$ from the standard basis $\mathbf{I}$ to $\mathbf{d}'$ in the basis $\mathbf{I} \circ \mathbf{k}$. Note that the scaling $\mathbf{k}$ also characterizes the resulting basis. The inverse scaling is

$$\mathbf{d} = \mathbf{d}' \cdot (\mathbf{I} \circ \mathbf{k})^{-1}. \tag{12}$$

Another scaling $\mathbf{k}^b$ gives the data record $\mathbf{d}''$ analog as

$$\mathbf{d}'' = \mathbf{d} \circ \mathbf{k}^b = \mathbf{d} \cdot (\mathbf{I} \circ \mathbf{k}^b) \tag{13}$$

in the basis $\mathbf{I} \circ \mathbf{k}^b$. By putting (12) in (13) follows the definition of a scaling between any basis $\mathbf{k}$ to any basis $\mathbf{k}^b$ with

$$\mathbf{d}'' = \mathbf{d}' (\mathbf{I} \circ \mathbf{k})^{-1} (\mathbf{I} \circ \mathbf{k}^b). \tag{14}$$

We apply rules for algebra, which gives for (14)

$$\mathbf{d}'' = \mathbf{d}' \cdot (\mathbf{I} \circ \mathbf{k}'), \text{ with} \tag{15}$$

$$\mathbf{k}' = (k_1^b / k_1, ..., k_n^b / k_n)^T. \tag{16}$$

This gives an interpretation of a scaling operation as follows: Each scaling is based on a related basis and equates to a transformation into another. In this work, we consider the standard basis $\mathbf{I}$ as related basis, i.e., $k_i = 1$ applies in (16), which simplifies (14) to (11) and thus gives the definition of a scaling operation by (11) and the definition of a scaling by $\mathbf{k} = (k_1, ..., k_n)$. The scaling $\mathbf{k}$ affects a shift of $\mathbf{d}$ to the novel position $\mathbf{d}'$ to the basis $\mathbf{I} \circ \mathbf{k}$. Figure 4 provides an illustration for the effect of shifting data points $\mathbf{d}$ by a scaling (k_1, k_2, k_3) of the standard basis $(1, 1, 1)$ for two different scaling operations with $(0.5, 1, 1)$ and $(2, 1, 1)$ for the case $n = 3$.

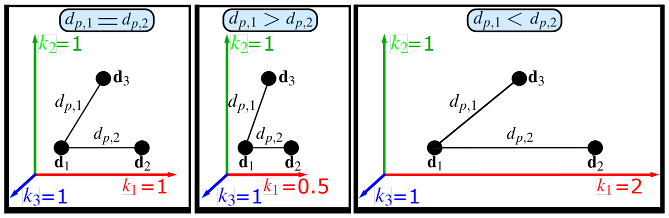

Fig. 4. The Scaling Issue: pairwise distances $d_{p,i}$ change with scaling.

5.2 How the Scaling Influences the Data Properties

Why is the *choice of scaling* important? Let us recap that the main goal of the (visual) data analysis is to reveal characteristic patterns of the data. Unfortunately, both the shape and occurrence of these patterns strongly depend on the choice of scaling. To explain why the scaling influences the structure of patterns, let us take a look at the general scaling-related Minkowski distance d_p between two data records $\mathbf{d}_i$ and $\mathbf{d}_j$, given by:

$$d_p(\mathbf{d}_i, \mathbf{d}_j, \mathbf{k}) = \sqrt[p]{\sum_{d=1}^{n} |d_{i,d} \cdot k_i - d_{j,d} \cdot k_j|^p} \qquad (17)$$

where $\mathbf{k} = \mathbf{1}$ and $p = 1$ gives the well-known p_1-norm aka Manhattan distance, $p = 2$ gives the p_2-norm aka Euclidean distance and so on. It can be seen that these distance measures depend on the level of scaling $\mathbf{k} = (k_1, \ldots, k_n)$. Figure 4 illustrates this effect: $d_{p,1} = d_{p,1}(\mathbf{d}_1, \mathbf{d}_3, \mathbf{k})$ is the distance between the records $\mathbf{d}_1$ and $\mathbf{d}_3$, while $d_{p,2} = d_{p,2}(\mathbf{d}_1, \mathbf{d}_2, \mathbf{k})$ gives the distance between $\mathbf{d}_1$ and $\mathbf{d}_2$. For the standard basis applies $d_{p,1} = d_{p,2}$. After scaling with $k_1 = 0.5$ and $k_1 = 2$ the distance relations completely change to $d_{p,1} > d_{p,2}$ and $d_{p,1} < d_{p,2}$, respectively. Thus, all characteristics of the data that are related to distances are scaling-dependent, such as the structure of the patterns, the clustering, the distribution and density estimation of the data: All these characteristics change if the scaling does. A well-known example for this effect is the change of the analysis results of the popular principal component analysis (PCA) [44] or of the multidimensional scaling (MDS) [42] under a varying scaling of the data. Thus, considering the scaling of data is mandatory for a successful data analysis.

Finally, we give a practical qualitative example for the scaling issue: Figure 1 illustrates how different scalings influence the data patterns for two synthetic datasets with $n = 3$ dimensions and $m = 1000$ data records, generated by [2] (the data can be found in the additional material). At the top, Star Coordinate projections for the dataset 1 are given: On the left, a projection is given for the dimension-wise choice of scaling $\mathbf{k} \approx (0.45, 2.3, 1.0)$. The data pattern shows a (weak) multi-linear correlation. For the next projection the choice of scaling of the data has changed to $\mathbf{k} \approx (1.7, 1.0, 0.7)$ and we can see that the structure of patterns changed dramatically. Now, the data contains two clusters while the correlation seems to have vanished, showing that the influence of scaling might dominate the behavior of the data completely. On the bottom, dataset 2 is discussed. The Star Coordinate projection on the left shows the data for a scaling with $\mathbf{k} \approx (0.43, 1.58, 1.0)$. The data contains two clear clusters. Changing the scaling to $\mathbf{k} \approx (1.45, 0.71, 1.0)$ leads to three clusters in the data. It also illustrates how the change of scaling strongly influences the pattern configuration.

5.3 Scaling: The Common Practice

Even though the scaling issue might be known, scaling operations are nonetheless often applied in practice, mostly without reflecting the negative effects of it. The most common reasons are

- (i) to convert data units,
- (ii) to convert the granularity of data units, or
- (iii) to normalize the data.

For instance for (i) a data dimension d with the unit degree *Fahrenheit* is converted into the degree *Celsius* equating to the scaling operation

$$d' = (d - 32) \cdot 0.5\overline{5}.$$

An example for (ii) is to convert the unit *kilogram* into *gram* by

$$d' = d \cdot 1000.$$

Or finally (iii), the data is normalized to the $0 - 1$ interval by

$$d' = (d - d^{\min}) \cdot \frac{1}{(d^{\max} - d^{\min})},$$

which is automatically done in a number of systems when the data is loaded. In doing so, the data analysis might reflect the scaling more than the data properties. Thus, we subsequently discuss how an informative scaling might be found.

6 Theory of the *LloydRelaxer*

In this section, we present both the theory on how scaling effects (=scaling issue) can be automatically minimized within a geometrical approach and aspects being relevant for the numerical handling and implementation of our concept. We start with the presentation of the underlying theory.

6.1 Formal Problem Setting

We now assume an affine transformation of the i-th dimension in the data space, which is defined by a scaling factor k_i and translation e_i, such that for every data point $\mathbf{d}_j$ the i-th dimension $d_{i,j}$ is transformed to

$$d'_{i,j} = k_i \cdot d_{i,j} + e_i \qquad \text{for} \qquad j = 1, ..., m. \qquad (18)$$

Since every dimension should be scaled and translated independently, we define all scalings and translations by the n-D vectors

$$\mathbf{k} = (k_1, ..., k_n)^T \quad , \quad \mathbf{e} = (e_1, ..., e_n)^T. \qquad (19)$$

Then, we can write the scaled and translated data matrix

$$\mathbf{D}' = (d'_{i,j}) = \mathbf{D} \circ (\mathbf{k} \cdot \mathbf{1}_m{}^T) + \mathbf{e} \cdot \mathbf{1}_m{}^T. \qquad (20)$$

The problem to be solved in this paper can now be stated as follows: given $\mathbf{D}$ and $\mathbf{A}$, find the optimal $\mathbf{k}$ and $\mathbf{e}$ such that Star Coordinates of the data set $\mathbf{D}'$ give the most informative projection.

Let $\mathbf{P}'$ be the matrix of all projected data points $\mathbf{D}'$. This gives

$$\mathbf{P}' = \mathbf{A} \cdot \mathbf{D}' = \mathbf{A}' \cdot \mathbf{D} + \mathbf{E}' \qquad (21)$$

with

$$\mathbf{A}' = \mathbf{A} \cdot \mathbf{K} \quad , \quad \mathbf{E}' = \mathbf{A} \cdot \mathbf{e} \cdot \mathbf{1}_m{}^T \qquad (22)$$

where $\mathbf{K} = diag(\mathbf{k})$, i.e., $\mathbf{K}$ is the diagonal matrix containing the values of $\mathbf{k}$. (21), (22) is obtained from (20) by applying elementary rules for matrix multiplication. It reveals the following properties: Firstly, applying a translation in one or more dimensions in the data space results only in a translation in the projection, i.e., it does not change the positions of the projected points relative to each other. Because of this, we do not consider translations of the dimensions, i.e, we assume $\mathbf{e} = \mathbf{0}_n$. Secondly, the matrix $\mathbf{A}'$ has an interpretation in Star Coordinates: if $\mathbf{A}$ is considered as the set of all coordinate axes in 2D, then $\mathbf{A}'$ is obtained by scaling the i-th coordinate axis by k_i for $i = 1, ..., n$. In other words: scaling the dimensions in data space is equivalent to scaling the coordinate axes in Star Coordinates. Consequently, the scaling parameters $\mathbf{k}$ in (9), (11), and (17) are all the same parameter, meaning that the scaling in the projection, data, and their influence to pairwise distances are directly interlocked and related to each other.

This allows us to formulate our problem as follows: given the data $\mathbf{D}$ and a normalized projection matrix $\overline{\mathbf{A}}$ (cf. (5)), find an optimal scaling $\mathbf{k}$ of the n coordinate axes such that Star Coordinates give the most informative projection.

6.2 Informative Projections

Informally spoken, an informative projection is a projection showing structures that are inherent properties of the high-dimensional data and not due to an arbitrary data scaling. We follow [13] and prefer projections with point layouts that are as regular as possible. In fact, [13] considers the Delaunay triangulation of the projected points and proposes several measures of their regularity. Unfortunately, they cannot be directly used for our problem: Since changing the scaling of the coordinate axes may lead to flips in the Delaunay triangulation, the measures are discontinuous under scaling the coordinate axes. While finding the minimum in the case of scatterplots (having only one degree of freedom) is possible (see [13]), it cannot be conveyed to the case of Star Coordinates with $n-1$ degrees of freedom. Because of this, we consider the dual structure that shows continuous changes under axes scaling: the Voronoi diagram. In fact, we search for a scaling of the coordinate axes such that the Voronoi diagram is as regular as possible.

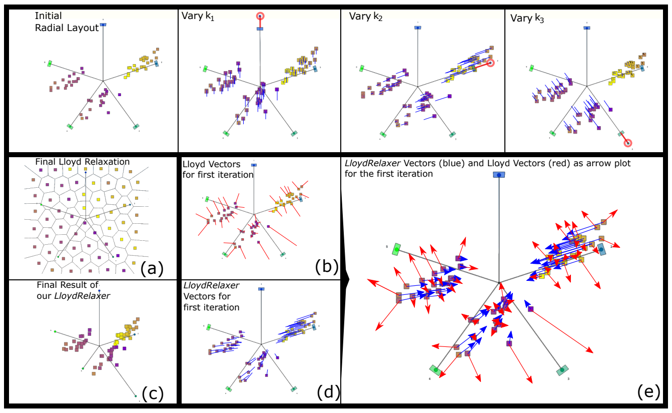

Fig. 5. Scheme of the theory for our *LloydRelaxer*: (top) Shift vectors for different scaling operations, (a) Result of a Lloyd Relaxation, (b) Shift vectors for the first step of a Lloyd Relaxation, (c) Result of our *LloydRelaxer*, (d) Shift vectors for the first change of scaling by **dk** of our *LloydRelaxer*, (e) Overlay view for (b) and (d).

Let $\mathbf{L}$ be the Lloyd matrix of $\mathbf{P}$ (cf. Fig. 6(b)). In general, a step of the Lloyd relaxation changes the locations of the points in a way that cannot be achieved by changing the scaling $\mathbf{k}$ of the coordinate axes only (cf. Fig. 6 (top)). Instead, we search for a scaling such that the change of the location of the projected points comes as close as possible to the change under one step of the Lloyd relaxation (cf. Fig. 6 (d-e)). Let $\mathbf{dk} = (dk_1, ..., dk_n)^T$ be the nD vector of the desired changes of scaling $\mathbf{k}$, i.e., the scaling of the coordinate axes changes from $\mathbf{k}$ to $\mathbf{k} + t \cdot \mathbf{dk}$ for a small step size t. This induces a change of the location of projected points from

$$\mathbf{P} = \overline{\mathbf{A}} \cdot \mathbf{K} \cdot \mathbf{D} \tag{23}$$

to

$$\mathbf{P}_{\mathbf{dk}} = \overline{\mathbf{A}} \cdot diag(\mathbf{k} + t \cdot \mathbf{dk}) \cdot \mathbf{D}. \tag{24}$$

Then, we search for the unknown $\mathbf{dk}$ such that $\frac{\mathbf{P}_{\mathbf{dk}} - \mathbf{P}}{t}$ comes as close as possible to $\mathbf{L}$, i.e., we solve

$$\left\| \frac{(\mathbf{P}_{\mathbf{dk}} - \mathbf{P})}{t} - \mathbf{L} \right\|_F^2 \quad \rightarrow \quad \min. \tag{25}$$

Since

$$\frac{(\mathbf{P}_{\mathbf{dk}} - \mathbf{P})}{t} = \overline{\mathbf{A}} \cdot diag(\mathbf{dk}) \cdot \mathbf{D}, \tag{26}$$

with (23) and (24), the system (25) is a quadratic function in $\mathbf{dk}$. Its solution is a linear system in $\mathbf{dk}$ that can be written as

$$\mathbf{S} \cdot \mathbf{dk} = \mathbf{w} \tag{27}$$

and thus

$$\mathbf{dk} = \mathbf{S}^{-1} \cdot \mathbf{w} \tag{28}$$

with

$$\begin{aligned} \mathbf{S} &= (\mathbf{D} \cdot \mathbf{D}^T) \circ (\overline{\mathbf{A}}^T \cdot \overline{\mathbf{A}}), & (29)\\ \mathbf{W} &= \overline{\mathbf{A}}^T \cdot \mathbf{L} \cdot \mathbf{D}^T, & (30)\\ \mathbf{w} &= diagv(\mathbf{W}), & (31) \end{aligned}$$

where *diagv* denotes the vector of the elements on the main diagonal of a quadratic matrix. The proof that (27)–(31) is the solution of (25) is provided in an accompanying Maple sheet. Then, one iteration step towards a most regular scaling is given with

$$\mathbf{A} = \overline{\mathbf{A}} \circ (\mathbf{k} + t \cdot \mathbf{dk}) \tag{32}$$

parametrized by the step size t. Following this iteration until $\mathbf{A}$ converges gives the final most regular and informative projection (cf. Fig. 6 (c)), i.e., our *LloydRelaxer* result. This projection is as simmilar as the data allows under scaling w.r.t. the plain result of the Lloyd Relaxation (cf. Fig. 6(a)).

The construction of $\mathbf{L}$ from $\mathbf{P} = \mathbf{A} \cdot \mathbf{D}$ is an elementary step to calculate $\mathbf{dk}$. This and further aspects regarding the numerical handling and implementation will be subsequently discussed.

7 Implementation

In this section, we discuss aspects of the implementation and we describe the final Algorithm 1 of our *LloydRelaxer* in detail.

7.1 Constructing Voronoi Centroids and Lloyd Matrix

We define the related centroids and Lloyd vectors (cf. 6 (a) right) of the Voronoi diagram (cf. 6 (a) left). For each Voronoi cell $\mathbf{v}(\mathbf{p})$ to each projected point $\mathbf{p}$ the 2D centroid $\mathbf{c}(\mathbf{p})$ (cf. 6 (b_{1-3})) is

$$\mathbf{c}(\mathbf{p}) = \int_{\mathbf{v}} \mathbf{p} \cdot p(\mathbf{p})\, d\mathbf{p},$$

with $p(\mathbf{p})$ equating to the density. Here, the density being $p(\mathbf{p}) = const$. Given a triangulation of the convex polygon (being the Voronoi cell) between point $\mathbf{p}$ and the Voronoi cell's corner points $\mathbf{p}_i$ (cf. 6 (c_{1-3})), this equation to calculate the centroid $\mathbf{c}(\mathbf{p})$ simplifies to:

$$\mathbf{c}(\mathbf{p}) = \frac{\sum_{i=1}^{w} \overline{\mathbf{p}}_i \cdot A_i}{\sum_{i=1}^{w} A_i}, \text{ with} \tag{33}$$

$$\begin{gathered} \overline{\mathbf{p}}_i = \frac{1}{3} \cdot (\mathbf{p} + \mathbf{p}_i + \mathbf{p}_{i+1}),\\ p = ||\mathbf{p}_{i+1} - \mathbf{p}_i||, p_i = ||\mathbf{p}_i - \mathbf{p}||, p_{i+1} = ||\mathbf{p}_{i+1} - \mathbf{p}||\\ s = \frac{1}{2} \cdot (p + p_i + p_{i+1}),\\ A_i = \sqrt{s \cdot (s-p) \cdot (s-p_i) \cdot (s-p_{i+1})}. \end{gathered}$$

The $2 \times m$ Lloyd matrix $\mathbf{L}$ contains the difference vectors between the projected points $\mathbf{p}_i \in \mathbf{P}$ and the Voronoi centroids $\mathbf{c}_i \in \mathbf{C}$,

$$\mathbf{L} = \mathbf{C}(\mathbf{P}) - \mathbf{P}. \tag{34}$$

7.2 Handling Doublings in the Point Set

The calculation of the Lloyd vectors $\mathbf{L}$ is a crucial step and thus requires to minimize all sources of errors. Regarding this, the underlying Voronoi diagrams can be generated for distinct point sets only, i.e., for point sets that do not contain a copied 2D position: $\mathbf{p}_i \neq \mathbf{p}_j, i \neq j$. If there are at least two points sharing the same position (doubling) $\mathbf{p}_i = \mathbf{p}_j$, two distinct Voronoi cells cannot be found for them in order to spatially separate these points. In our approach, there are two situations where such doublings might occur.

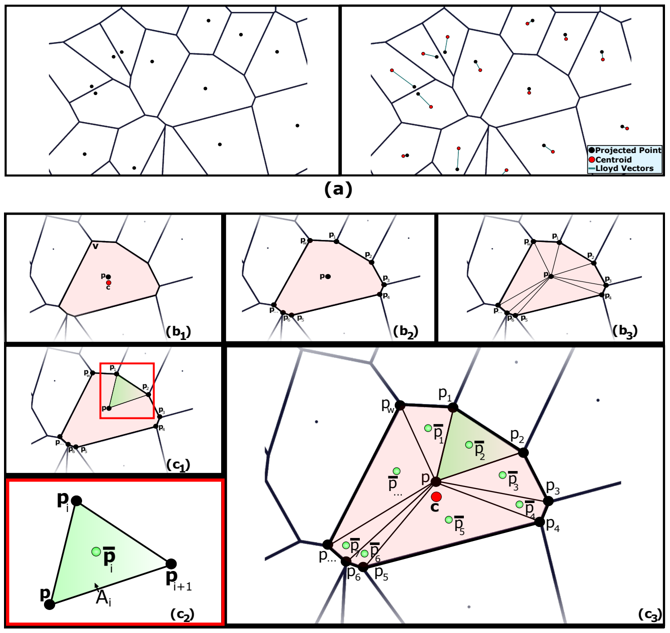

Fig. 6. Centroids and Lloyd vectors: (a) Voronoi diagram (left) and Centroids (right), (b-c) triangulation of a Voronoi cell and its characteristics.

1.) First, double data values $\mathbf{d}_i = \mathbf{d}_j \in \mathbf{D}$ must produce a doubling $\mathbf{p}_i = \mathbf{p}_j \in \mathbf{P}$ in the projection, since it applies:

$$(\mathbf{d}_i = \mathbf{d}_j) \rightarrow ([\mathbf{p}_i = \mathbf{A}\cdot \mathbf{d}_i] \wedge [\mathbf{p}_j = \mathbf{A}\cdot (\mathbf{d}_j = \mathbf{d}_i)]) \rightarrow (\mathbf{p}_i = \mathbf{p}_j).$$

To handle this first case, we remove all double data values in the data matrix $\mathbf{D}$ before the *LloydRelaxer* is applied. This is justified by the fact that the general geometric properties of the related 2D point set appear similar – with or without these double values – and independent of the projection matrix $\mathbf{A}$.

2.) The second case happens if a projection $\mathbf{A}$ projects different data points $\mathbf{d}_i \neq \mathbf{d}_j$ onto the same projected points $\mathbf{p}_i = \mathbf{p}_j$:

$$[(\mathbf{p}_i = \mathbf{p}_j)|(\mathbf{p}_i = \mathbf{A}\cdot \mathbf{d}_i) \wedge (\mathbf{p}_j = \mathbf{A}\cdot \mathbf{d}_j)] \rightarrow (\mathbf{d}_i \neq \mathbf{d}_j).$$

This case cannot be trivially handled and is a singularity configuration. Deleting the data $\mathbf{d}_i$ or $\mathbf{d}_j$ is not an option, since the related doubling $\mathbf{p}_i = \mathbf{p}_j$ depends on and is caused by the projection matrix $\mathbf{A}$, and is not structurally stable, i.e., it would disappear by slightly changing $\mathbf{A}$. In addition, deleting $\mathbf{d}_i$ or $\mathbf{d}_j$ is also not an option, since this would change the semantic information of the data and thus the data itself. For this case, our approach has to preserve the data $\mathbf{d}_i$ and $\mathbf{d}_j$ to maintain the ability to build up a Voronoi diagram, which means that our approach has to recover $\mathbf{p}_i \neq \mathbf{p}_j$: For this, the projection matrix $\mathbf{A}$ could be slightly varied. However, this is not a good idea, since it would be a global operation changing the positions of all points, which might cause even more second cases in the projected point set. Thus, it is advisable to reposition solely one of the two points to the position $\mathbf{p}'_i$ by adding some noise: $\mathbf{p}'_i = \mathbf{p}_i + \mathbf{n}$ and $\mathbf{n} = t_1 \cdot (\cos(t_2)\ \sin(t_2))^T$, with $t_1 \in (0, 0.01]$ and $t_2 \in [0, 2\cdot\pi]$ being random numbers, where t_1 controls the distance of the shift and t_2 controls the direction for this repositioning. Thus, the handling of this second case introduces a small error related to the intensity of the added noise $\mathbf{n}$ in order to resolve this singularity. However, this second case has never been observed within our tests with real data (but in our synthetic data which we designed to reveal this effect). This is plausible, because it is numerically unlikely to accidentally meet a projection $\mathbf{A}$ that projects arbitrary data to the same points.

Beyond these two cases for a doubling, due to numerical reasons, a *quasi doubling* might occur. A quasi doubling means that the distance (e.g., Euclidean or Manhattan) between pairwise projected points $\mathbf{p}_i$ and $\mathbf{p}_j$ is so small that further mathematical operations for the remaining *LloydRelaxer* process become numerically unstable and fail, i.e.,

$$||\mathbf{p}_i - \mathbf{p}_j|| < \tau.$$

The lower numerical boundary of τ is 10^{-38} for float and 10^{-308} for double value computation operations. Practically and from experience τ should be chosen $\tau \approx 0.001$. To resolve such quasi doublings in the projected data $\mathbf{P}$, the distances between $\mathbf{p}_i$ and $\mathbf{p}_j$ can be increased to be larger than τ by applying a uniform multiplication k_n to the projected points $\mathbf{P}$. From experience, $k_n = 100$ yields convincing results. The Lloyd vectors $\mathbf{L}$ suppressing quasi doublings can then be computed by exchanging (34) with:

$$\mathbf{L} = (\mathbf{C}(\mathbf{P}\cdot k_n) - \mathbf{P}\cdot k_n)/k_n. \tag{35}$$

7.3 Handling of Border Voronoi Cells

In practice, the Voronoi diagram has to be bordered by a rectangle with the diagonal $\mathbf{b}_{\min} = (b_{x_{\min}}\ b_{y_{\min}})$ and $\mathbf{b}_{\max} = (b_{x_{\max}}\ b_{y_{\max}})$, giving four border edges. We define a *border Voronoi cell* as a Voronoi cell that touches the boundary, i.e., that shares at least one edge with the border of the Voronoi diagram. The remaining Voronoi cells are denoted as *inner Voronoi cells*. The Lloyd vectors being related to border Voronoi cells do not reflect the data but the choice of borders. Thus, depending on the chosen boundary, these Lloyd vectors might have an arbitrary Euclidean length, which is unrelated to the data.

Figure 7 (top) illustrates a typical scenario for the evolution of a Voronoi cell while the projection matrix $\mathbf{A}$ is being changed smoothly: Here, 12 samples of the time series are presented. During the evolution, the marked Voronoi cell starts as an inner Voronoi cell (red) and later becomes a border Voronoi cell (green) and vice versa. The profile of this Voronoi cell for the length of the related Lloyd vectors for this time series in Figure 7 (top) illustrates that the Lloyd vector becomes twice as long as usual during the border Voronoi phase of this cell. Please note that, depending on the choice of the border, they could be even longer. Figure 7 (bottom) illustrates this effect for a Voronoi diagram of a typical point set configuration. The Lloyd vectors of the border cells are a couple of times longer than usual, which is stressed in the illustration for the distribution of the length of the Lloyd vectors. Fortunately, the number of border Voronoi cells is usually small. Here, it is 13 out of 147 cells, which equates to a low rate of 8%. This is a usual relation. In addition, the number of inner Voronoi cells grows much faster with growing number m of points than the number of border Voronoi cells does. In total, a small relation/number of border Voronoi cells remains, which might have long and data-unrelated Lloyd vectors, strongly influencing the calculation for the *LloydRelaxer*. In order to handle this, the Lloyd vectors $\mathbf{l}_i \in \mathbf{L}$ of the border Voronoi cells are not considered within the Lloyd matrix $\mathbf{L}$, i.e, they are set to $\mathbf{0}$ in $\mathbf{L}$.

7.4 Limiting the Lloyd-based Expansion

The Lloyd relaxation only converges if a certain limit in space is considered, such as a boundary box. Thus, we limit the maximum length of an anchor point $\mathbf{a}_i = (x_i\ y_i)^T$ to one $||\mathbf{a}_i|| = 1$ by defining an additional uniform scaling k: $(\mathbf{k} + t\cdot \mathbf{dk})\cdot k$ w.r.t. (32). From this, k directly follows as:

$$k = \sqrt{1/(x_i^2 + y_i^2)},\ \text{where i relates to}\ \max(\sqrt{x_i^2 + y_i^2}). \tag{36}$$

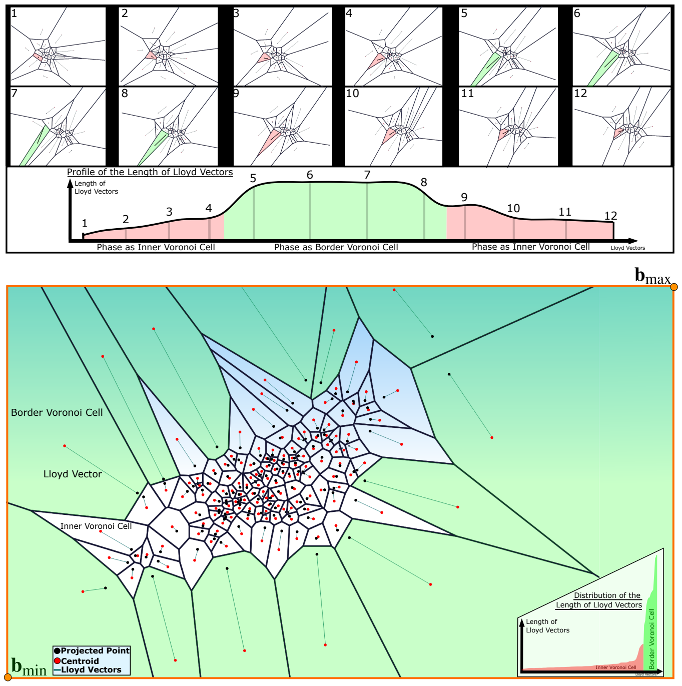

Fig. 7. Border Voronoi Cells vs. Inner Voronoi Cells: the Eucledian norm of the Lloyd vectors is unrelated to the data but related to the choice of border for the border Voronoi cells. (top) Observation of the evolution of the colored Voronoi cell during its life time under a smooth transition of the projection, (bottom) typical configuration of border Voronoi cells. Here, 13 border and 134 inner Voronoi cells occur.

Figure 8 shows how such an axis point configuration is limited by a uniform multiplication with k. Note that the pairwise aspect ratio k_i/k_j is untouched by applying k, i.e., $k_i/k_j = (k\cdot k_i)/(k\cdot k_j)$.

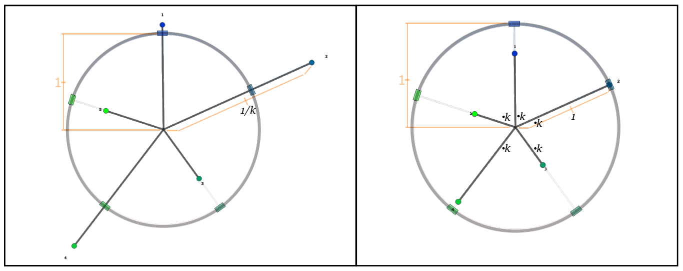

Fig. 8. Restrict an anchor point configuration into the unit circle for $n = 5$: (left) arbitrary configuration, (right) restricted configuration to unit circle by applying a multiplication with k to the anchor points.

7.5 The LloydRelaxer in a Nutshell

Here, we assemble the mentioned modules from Sec. 6.2 to 7.4 in one scheme, which gives the Algorithm 1, i.e., the *LloydRelaxer*.

The input of the algorithm is a normalized projection matrix $\overline{\mathbf{A}}$, an initial set of scaling factors $\mathbf{k}$, and the data matrix $\mathbf{D}$. $\overline{\mathbf{A}}$ defines the projection directions on the unit circle and it is the only required input from the user. For the initial $\mathbf{k}$, we use the unit scaling $\mathbf{k} = \mathbf{1}^T = (1,...,1)^T$, since we are interested in finding the aspect ratios k_i/k_j that minimize scaling effects, which does not depend on the level of an initial uniform scaling. $\mathbf{D}$ is the data where all double values are removed to handle the first case for doubling as explained in Sec. 7.2.

The algorithm requires a set of further parameters: $it_{\max}$, t, ε, $\mathbf{B}$, k_n, τ. The maximum iteration number $it_{\max}$ guarantees that the algorithm ends up in finite time. Step size t influences the speed of converging, while the threshold ε characterizes what the best possible quality of converging could be, i.e., $\varepsilon = 0$ is optimal. The border parameter $\mathbf{B} = (\mathbf{b}_{\min},\ \mathbf{b}_{\max})$

Algorithm 1 The LloydRelaxer

```
procedure LLOYDRELAXER($\overline{\mathbf{A}}$, $\mathbf{k}$, $\mathbf{D}$, $it_{\max}$, $t$, $\varepsilon$, $\mathbf{B}$, $k_n$, $\tau$)
$\mathbf{S}^{-1} = (\ (\mathbf{D}\cdot\mathbf{D}^T) \circ (\overline{\mathbf{A}}^T \cdot \overline{\mathbf{A}})\ )^{-1}$, $\mathbf{dk} = \infty$, $it = 0$, $\mathbf{A} = \overline{\mathbf{A}} \circ \mathbf{k}$

//Minimize the scaling effects by iteratively optimizing k.
while ( $(||\mathbf{dk}|| > \varepsilon) \wedge (it \le it_{\max})$ ) do
    $\mathbf{P} = \mathbf{A}\cdot\mathbf{D}$

    //Handle doubling of the 2nd Case (cf. Sec. 7.2).
    for all $\mathbf{p}_i, \mathbf{p}_j \in \mathbf{P}, i \neq j$ do
        if $((\mathbf{p}_i - \mathbf{p}_j) == 0)$ then
            $t_1 = random(0, 0.01]$, $t_2 = random[0, 2\cdot\pi]$
            $\mathbf{p}_i = \mathbf{p}_i + t_1 \cdot (\cos(t_2)\ \sin(t_2))^T$
        end if
    end for

    //Generate the Lloyd Matrix by (34-35) of Sec. 7.1-7.3.
    if $(\min(\mathbf{p}_i - \mathbf{p}_j) < \tau)$ then
        $\mathbf{L} = (\mathbf{C}(\mathbf{P}\cdot k_n) - \mathbf{P}\cdot k_n)/k_n$
    else
        $\mathbf{L} = (\mathbf{C}(\mathbf{P}) - \mathbf{P})$
    end if

    //Build the new k to minimize scaling effects (cf. Sec 6.2).
    $\mathbf{w} = diag(\overline{\mathbf{A}}^T \cdot \mathbf{L}\cdot\mathbf{D}^T)$
    $\mathbf{dk} = \mathbf{S}^{-1}\cdot\mathbf{w}$
    $\mathbf{k} = \mathbf{k} + t\cdot\mathbf{dk}$

    //Limit the Lloyd expansion with $k$ from (36) (cf. Sec. 7.4).
    $\mathbf{k} = \mathbf{k}\cdot k$

    //Calculate one step towards the most informative view.
    $\mathbf{A} = \overline{\mathbf{A}} \circ \mathbf{k}$
    $it{+}{+}$
end while
return $\mathbf{A}, \mathbf{k}$
```

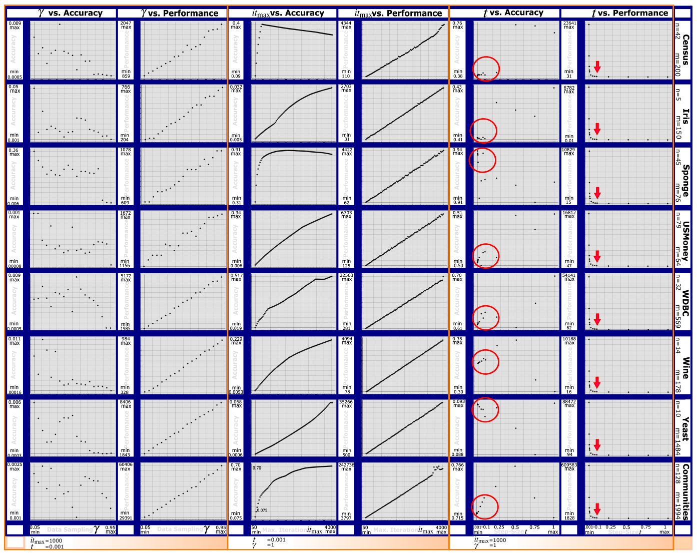

Fig. 9. Quantitative Evaluation: *accuracy* and *performance* behavior for varying parameters of the *LloydRelaxer*'s parameter space and different benchmark data.

defines the size of the Voronoi diagram. It should be as large as the border rectangle of the convex hull of the projected data points $\mathbf{p} = \mathbf{A}\cdot\mathbf{d}$. Because of (36), the largest component value in $\mathbf{A}$ is 1, which gives the upper estimation of the borders by $\mathbf{p} = \mathbf{1}\cdot\mathbf{d} = \sum_i^n d_i$ as $\mathbf{b}_{\min} = (-\max(\sum_i^n d_i), -\max(\sum_i^n d_i))^T$ and $\mathbf{b}_{\max} = (\max(\sum_i^n d_i), \max(\sum_i^n d_i))^T$. If the data is normalized, i.e., $d_i \leq 1$ the calculation of the upper estimation for the border simplifies to $\mathbf{b}_{\min} = (-n,-n)^T$ and $\mathbf{b}_{\max} = (n,n)^T$.

For handling quasi doublings, the parameters k_n and τ are required. From experience, $k_n = 1000$ is convenient. Since a uniform scaling does not affect the aspect ratios k_i/k_j, the lines 13 and 15 to 17 from Algorithm 1 could be ignored, meaning that the τ-based test to find out whether the handling of pseudo doubling is required will not be applied. Instead, the handling of pseudo doublings would be applied blindly by default. Then, the parameter τ is not required anymore.

Additionally, since it is numerically unlikely to get a doubling – and even if, it would just negatively influence one or two frames under an interaction with $\mathbf{A}$ – one could ignore the lines 5 to 11 from Algorithm 1 if one is willing to pay this price.

Let us take a look at the runtime of the algorithm. For this, we only need to focus on the dominate parts: Line 4 costs $\mathbf{O}(n\cdot m)$, line 13 to 17 cost $\mathbf{O}(m\cdot\log\cdot m)$ since they include Fortune's algorithm, line 19 is with $\mathbf{O}(n\cdot m^2)$ expensive, and from line 3 we know that the algorithm converges after a number of $it_{\max}$ iterations, at least. This gives the runtime by

$$(\mathbf{O}(n\cdot m) + \mathbf{O}(m\cdot\log\cdot m) + \mathbf{O}(n\cdot m^2))\cdot it_{\max} = \underline{\mathbf{O}(n\cdot m^2\cdot it_{\max})}.$$

We see that by using the complete m data records of $\mathbf{D}$ the algorithm performs worse the more m grows. In a statistical sense, the information of the data might also be contained and preserved within a subset, if m is large. Thus, in order to reduce the number m of used points, we apply a uniform data sampling $\mathbf{D}_\gamma$ analog to [25] and based on [34], with $\gamma \in [0,...,1]$ equating to the amount of used data, i.e., $\mathbf{D}_{\gamma=1} = \mathbf{D}$, for our Algorithm 1. In [25] it is proposed to generally use $\gamma = 0.1$, i.e., 10% of the data $\mathbf{D}$, but their application is different to ours. Here, from evaluation (cf. Sec. 8.1), about $\gamma = 0.2$ gives good results and performs well. In the next section, our approach will be evaluated.

8 Results / Evaluation

In this section, we provide a quantitative evaluation in order to analyze the relation between our algorithm and its parameterization. In addition, we give a qualitative evaluation to illustrate the abilities of minimizing scaling effects for the projection-based visual data exploration. The evaluation was conducted on a 64 Bit Dual Core Intel CPU 2,6 GHz mobile computer in single thread mode with an NVIDIA GeForce GTX 950M and OS Win 8.1.

8.1 Quantitative Evaluation

We conducted a set of experiments based on the standard radial layout of Star Coordinates for the projection matrix $\overline{\mathbf{A}}$ in order to measure the behavior of the parameters γ, t and $\text{it}_{\max}$ w.r.t. the *accuracy* and *performance* of the *LloydRelaxer*. We do so by varying a parameter of interest and holding the remaining parameters. For this, the *accuracy* a_e of the parameter $e \in \{\gamma, t, it_{\max}\}$ is the normalized Manhattan distance $d_1(\mathbf{k}_1, \mathbf{k}_r^e)$ (cf. (17)) between the initial scaling $\mathbf{k}_1 = (1,...,1)^T$ and the resulting scaling $\mathbf{k}_r^e$ after running the *LloydRelaxer* by varying parameter e as $a_e = |d_1(\mathbf{k}_1, \mathbf{k}_r^e, \mathbf{k}_1)|/n$. Those values for parameter e which produce the same a_e values give the same accuracy behavior. In contrast, the *performance* is measured with the run-time of the algorithm in milliseconds. Figure 9 presents the behavior of the parameters in e for 8 benchmark datasets, with $n \in (5,...,128)$ and $m \in (150,...,1994)$. Please see UCI repository [27] or [25], [26] for more details about the datasets. In total, we ran 880 experiments which cover the 8 datasets, with a portion of 152 runs in order to investigate the influence of the data sampling rate γ, 632 runs to choose an optimal maximum iteration number $it_{\max}$, and 192 runs to disclose the behavior regarding step size t.

Data Sampling γ: For parameter γ (Figure 9 (left)), we are interested in finding a rate which gives accurate results and performs well. The accuracy error stays within 10^{-3} if we chose $\gamma = 0.2$, except for the data Iris and Sponge. But these datasets have a low data record number m. Hence, we choose $\gamma \geq 0.2$ if $m \geq 150$ and $y \geq 0.8$ else. This way, we also get an acceptable performance with one outlier: the Community dataset. Here, a clustering would be a good possibility to improve the performance. However, this is far beyond the scope of this paper. Thus, we leave it to future work.

Step Size t: The performance exponentially grows with decreasing step size t (Figure 9 (right)). If $t \in (0.001,...,0.1)$, then the accuracy behavior is quite similar (red circles), but the performance gets exponentially worse if $t < 0.1$ (red arrows). Thus, t should be chosen close to $t = 0.1$.

Maximum Iteration Number $it_{\max}$: The iteration number is well chosen if the converging behavior for the accuracy is good, which is given by a small slope in Figure 9 (middle). For most of the cases, an iteration number of 4000 gives an appropriate convergence, and if not, the subsequent changes regarding the final scaling results are negligible. So 4000 steps are convenient. In this test, for reasons of error minimization we worked with a step size of $t = 0.001$. Note that the *LloydRelaxer* is an iterative process, i.e, if the step size is divided by f, the iteration number has to be multiplied by f in order to reach the same solution quality. This is a well-known relation between an iteration number and a step size. We found out above that a step size of $t = 0.1$ leads to good results. Thus, an iteration number of 40 $((4000\cdot 0.001)/0.1)$ is appropriate. In total, based on these tests, we recommend to choose $\gamma = 0.2$, $it_{\max} = 100$, $t = 0.05$ to guarantee good results if the *LloydRelaxer* should be used in an interactive mode. Otherwise, the parameters can be chosen with a better resolution which is subsequently presented for the qualitative evaluation.

8.2 Qualitative Evaluation of Synthetic 2D Data

In this section, we consider two-dimensional synthetic data [2] in order to qualitatively evaluate comprehensive properties of our approach and for reasons of completeness. For this, Figure 10 (top) shows a set of scatterplots of synthetic 2D datasets that contain three compact clusters and $m = 1000$ data records each. One cluster is shifted from the left to the right (green line) in order to "simulate" different configurations. Note that Star Coordinates and scatterplots are equivalent in the 2D case. Figure 10 (bottom) shows a set of scatterplots of synthetic datasets that contain two clusters this time. Again, one of these clusters is shifted from the left to the right. Finally, Figure 10 (bottom-left) shows an example for a complex pattern in the 2D dataspace (with $m = 10000$ data records). Obviously, to automatically select the aspect ratio for 2D data is the same as scaling one of the two available axes. So, the degree of freedom is only one.

For each sample in this Figure 10, the original scatterplots visualization (= label **Original**) and the resulting scatterplot, when applying our *LloydRelaxer*, is given (= label ***LloydRelaxer***). In addition, the figure shows two further variations of our *LloydRelaxer*: first, without handling the border Voronoi cells (= **without border handling**, cf. Section 7.3), and second, based on an alternative

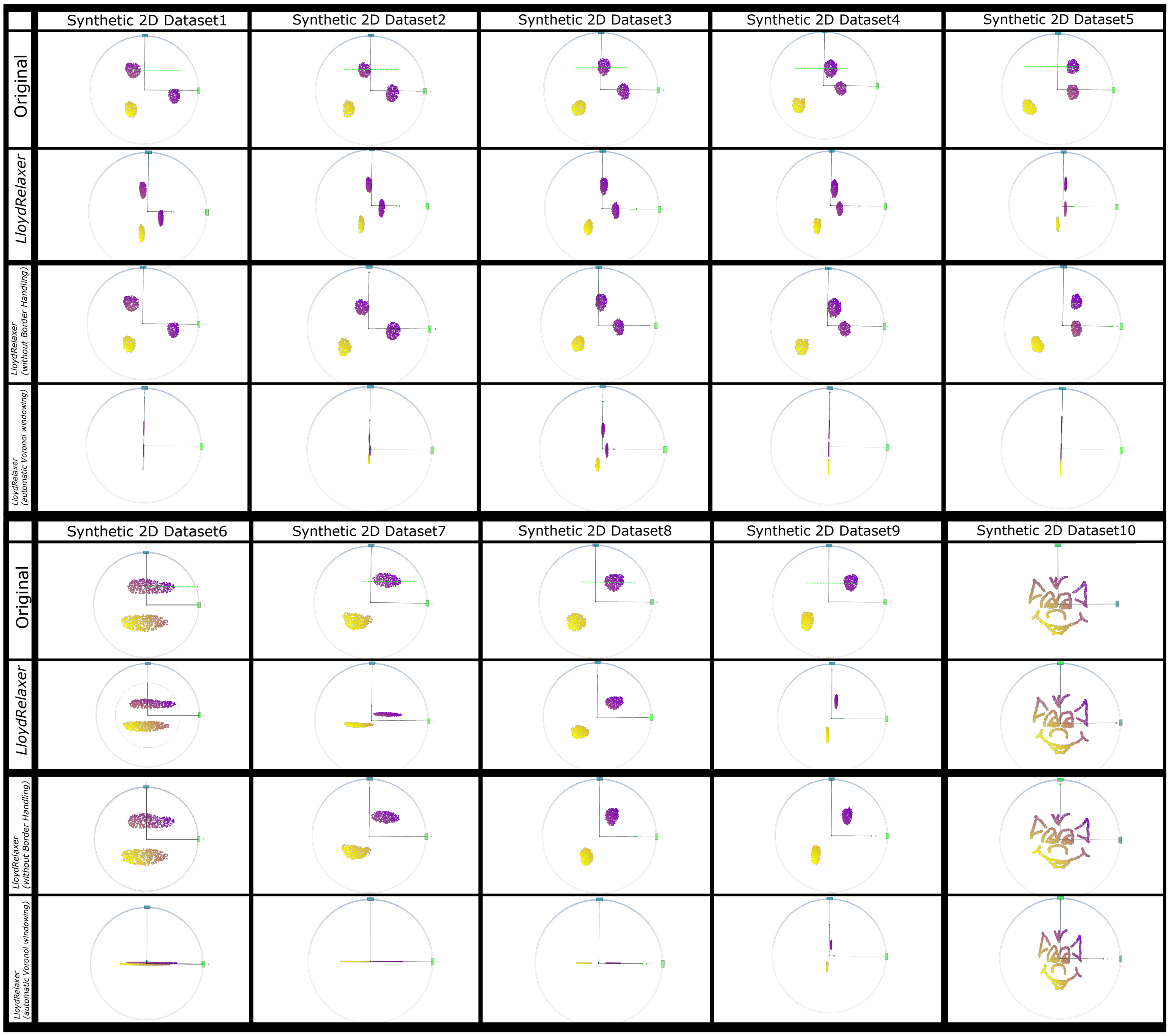

Fig. 10. Simple Qualitative Evaluation on synthetic data: The index of the data is color-coded. (Top-Box) A cluster in 2D data is moved along the green path within a dataset containing three clusters. (Bottom-Box). The same application as seen in the Top-Box, but with two clusters instead of three. Both boxes further illustrate the outcomes for the *LloydRelaxer* for these clustered data. (Bottom-Left) The outcome of our *LloydRelaxer* for a complex pattern is shown.

boundary handling of the Voronoi cells, called *automatic Voronoi windowing* (= **automatic Voronoi windowing**). Here, instead of using a fixed size to define the bounding box of the Voronoi diagram, given by the parameters $\mathbf{b}_{\min}$ and $\mathbf{b}_{\max}$, we consider the boundary border of the convex hull of the point set at each iteration step. This has the effect that the size of the Voronoi diagram constantly shrinks over the *LloydRelaxer* process.

Analysis: Figure 10 (top): It can be seen that the *LloydRelaxer* preserves the clustering of the data the more the less linear independent the structures are to each other; meaning, as long as they cannot be mutually revealed from each other by just playing around with the scaling of one dimension. Thus, Figure 10 (top-right) shows barely a linear structure, which fits the data best, while Figure 10 (top-left) shows the three clusters clearly separated from each other. The same applies to the example in Figure 10 (bottom): the middle figure shows the separated clusters, while the remaining examples rather stress the linearly related nature of the data. In addition, it can be seen that the border handling is important to the complete process. Without it, the border Voronoi cells (which are not related to the data) seem to dominate and thus to disturb the complete relaxing process, i.e., the results are almost equivalent to the original data. Lastly, the automatic Voronoi windowing illustrates that the complete structure would collapse without having a reasonable and constant choice of the size of the Voronoi diagram. But there is one exception: the complex pattern. The complex pattern (Figure 10 (bottom-right)) is linearly completely unrelated w.r.t. its data records in a way that – informally speaking – even such a collapsing behavior cannot make the pattern disappear. However, it follows that the *LloydRelaxer* has nevertheless to be parametrized w.r.t. the outcome of our quantitative evaluation.

Figure 11 shows a set of further experiments, with the above-mentioned 2D complex pattern (left) and the two multivariate 3D dimension reduction benchmark datasets *Spiral* [5] (middle) and

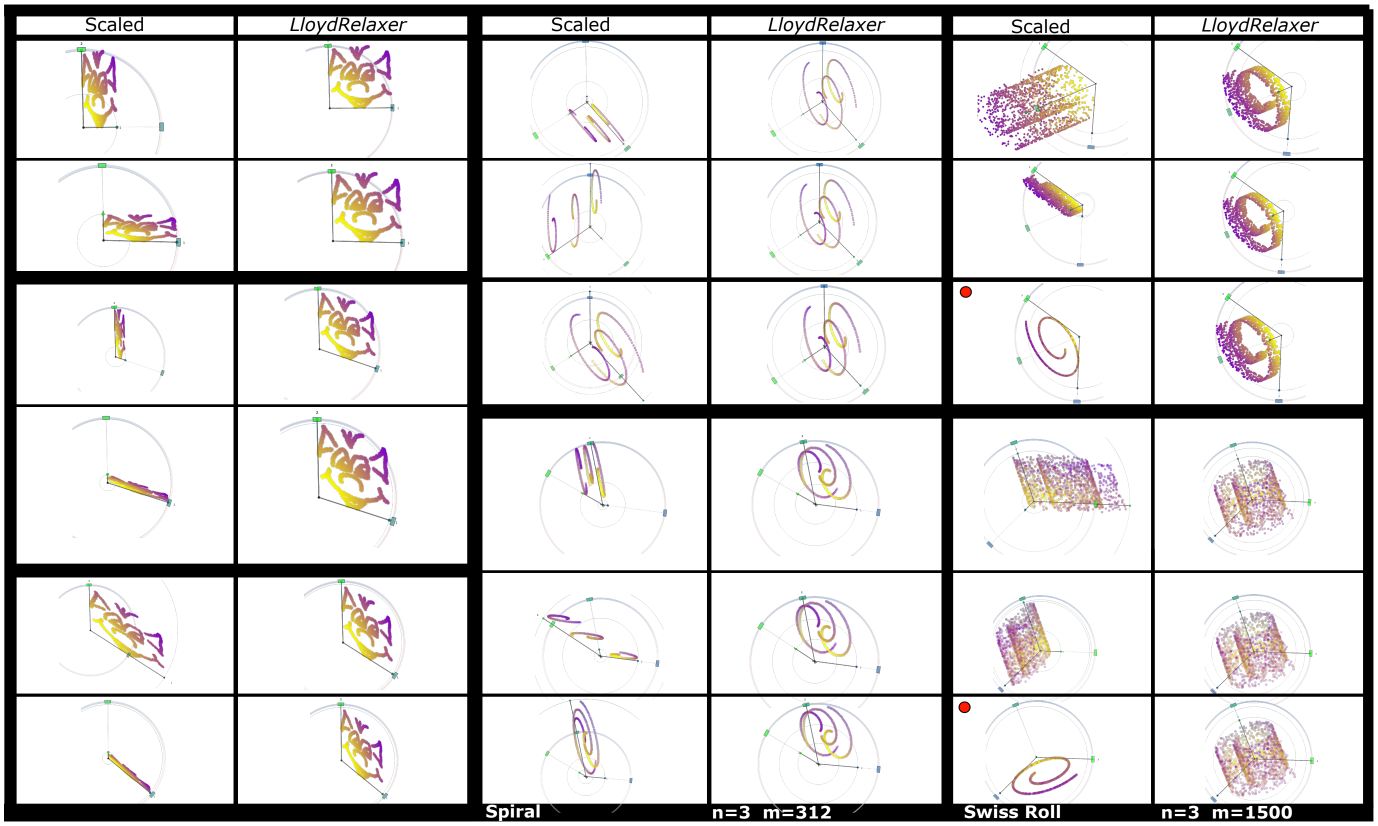

Fig. 11. Complex Qualitative Evaluation on synthetic data: The index of the data is color-coded. The column labeled with *scaled* illustrates different initial scalings for different benchmark data and the column labeled with *LloydRelaxer* illustrates the related outcome by our approach.

Swiss Roll [43] (right). For each case, arbitrary scaling configurations (= label *Scaled*) are used to initiate our *LloydRelaxer*. Then, two properties of our *LloydRelaxer* can be observed: (i) independent of the initial scaling, our *LloydRelaxer* produces the same output in our experiments, solely dependent on the chosen initial projection direction (i.e., the choice of $\overline{\mathbf{A}}$); (ii) global structures – i.e., structures that are related to all dimensions of the dataset (not only to a certain subspace) – are worked out and preferred. For instance, the *Swiss Roll* contains a sort of spiral that is swirling around a cylinder. A good projection should disclose this relation independent of the chosen projection direction. While the spiral structures just “live” in 2 of the 3 dimensions, the cylinder structures “lives” in all 3, which a good descaling should disclose and which our *LloydRelaxer* does for all cases (e.g, see Figure 11 (red points)). From the qualitative experiments with synthetic data, we can learn that global structures could be revealed efficiently by our *LloydRelaxer* by choosing appropriate aspect ratios automatically.

8.3 Qualitative Evaluation of Real nD Data

In this section, we conduct experiments with real data this time in order to observe how the above mentioned properties of our *LloydRelaxer* affects real-life applications. In Figure 12, we present a set of real data projection examples from the multi-dimensional datasets Census [27], WDBC [40], Yeast [32], Sponge [27], and the Abalone [45] in order to qualitatively evaluate and discuss the *LloydRelaxer*. For each sample, a projection with a pattern of interest can be seen (bottom-left). We utilized [26] to find such an interesting projection by having the underlying data normalized into the $0-1$ interval. In the bottom-right, the resulting scaling and projection can be seen for this pattern after our *LloydRelaxer* has been applied, parameterized with $t = 0.05$, $\gamma = 1$, and $it_{\max} = 1000$. On the top, the evolution during the *LloydRelaxer* algorithm of the scaling and projection matrix is visualized.

In the Census example, the complex original pattern (left) vanishes and two clearly separate clusters appear after applying our *LloydRelaxer* (right). It is also of interest that by far only one dimension contributes most to this cluster structure, while the remaining dimensions have much less (but not zero) influence on it. Thus, a linear separation for the two groups could be done easily, i.e., in this case a kind of visual classification or linear discrimination process would be supported (in case of need).

In the WDBC example, we see two global clusters in the projected data at first (left), which turns into a multi-linear relation (right) as most informative view. Here, medical doctors are usually interested in seeing the two “global” clusters, which allow to discriminate between malignant and benign breast cancer tumors. Does our result mean that there is no such discriminator? No, it does not. We also considered the ’ID’ variable here, which modifies the global structure, in order to illustrate that our *LloydRelaxer* really reveals the global pattern of the input data. Obviously, the considered variables need to be chosen carefully by an analyst in advance to avoid a further misleading source.

In the Yeast example, a collection of outliers is found (left), which are even preserved under our *LloydRelaxer*. Here, the initial scaling already shows an optimal global pattern. In the Sponge example, two clusters and some outliers can be seen (left). Finally, it turns out that four clusters – two larger clusters and two which are rather small (right) – can be distinguished and seem to be the better global pattern.

In the Abalone example, three clearly separated multivariate linear structures occur (left) at first, but after relaxing the scales these linear structures are not that well separated anymore. In fact, only one linear structure seems to be reliably well separated (right,

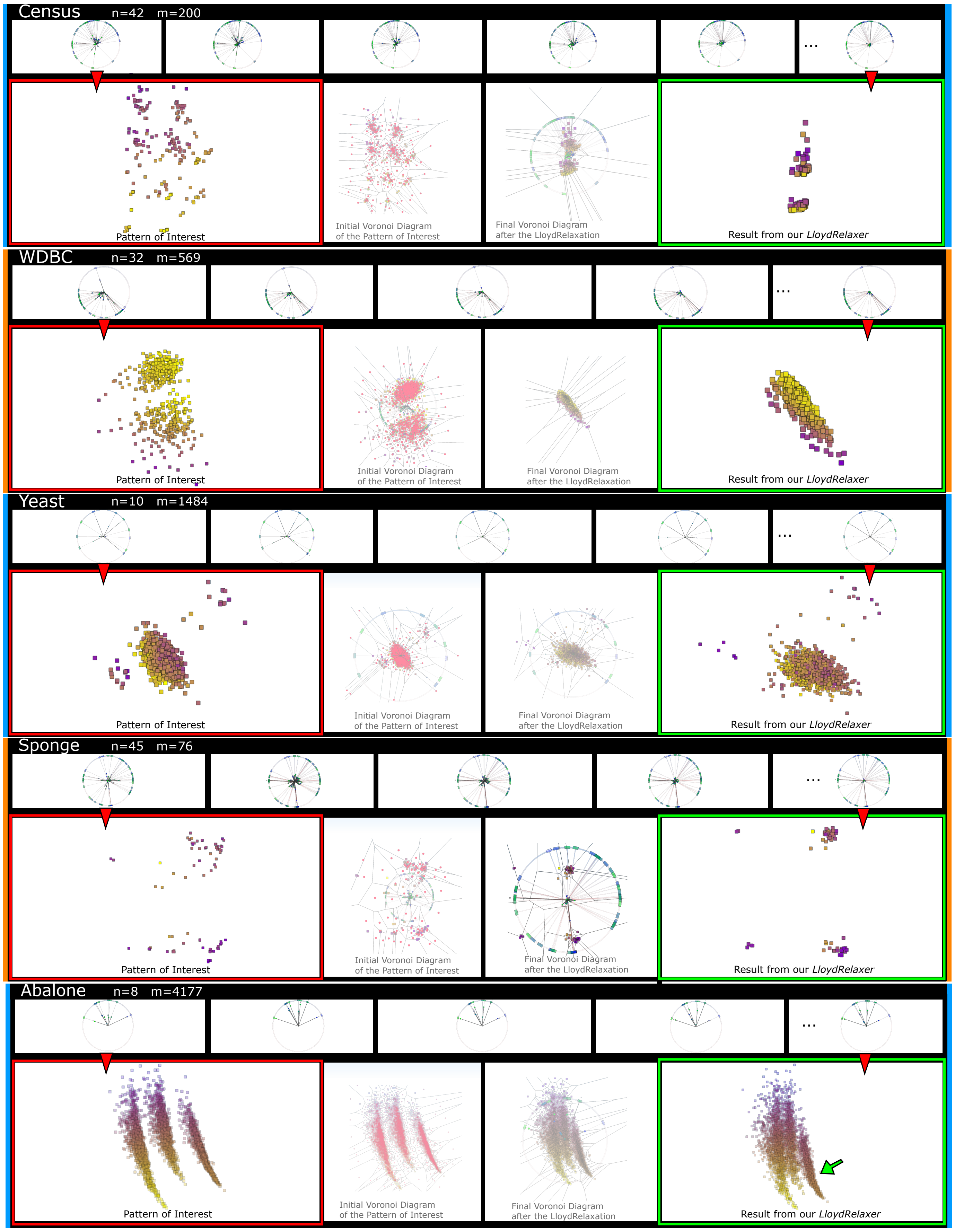

Fig. 12. Qualitative Evaluation on real data: The index of the data is color-coded. A projection of interest was selected for each data of a set of multi-dimensional benchmark dataset. This gives an initial scaling configuration (left, red border). On the right (green border), the descaled version of this projection after applying our *LloydRelaxer* is given. Structures that can be seen can be better explained by the data now, instead by scaling. Further information are given by the initial and final Voronoi diagram and the evaluation of the visualized projection matrices during our approach.

green arrow). However, both structures look quite similar, meaning these global structures are a result of the data, and not accidentally caused by scaling artefacts.

Our examples illustrate that the reliability w.r.t. patterns of the data which are seen within a projection can be increased by using our *LloydRelaxer*. Thus, to integrate the *LloydRelaxer* within a visual exploration process enables to robustly find relevant data patterns that are not only caused by scaling artefacts.

9 Discussion and Conclusion

There is an underrated aspect when analyzing data: scaling artifacts. They disturb the visual analysis. Thus, we presented the *LloydRelaxer* to address this scaling issue for multivariate projection of nD data. We do so by defining a smooth transition from the initial scaling to an as-regular-as-possible scaling via concepts from computational geometry, such as Voronoi diagrams and the Lloyd relaxation.

An Interactive Tool: We integrate our *LloydRelaxer* in an interactive tool, allowing a user-based visual exploration process. For this, the user interactively adopts directions in the matrix $\overline{\mathbf{A}}$ to get the traditional projections. In order to get our optimized scaling, our tool applies the Algorithm 1 on $\overline{\mathbf{A}}$ continuously after each interaction. To keep the process interactive, a data sampling is used when required, as already discussed in Section 7.5.

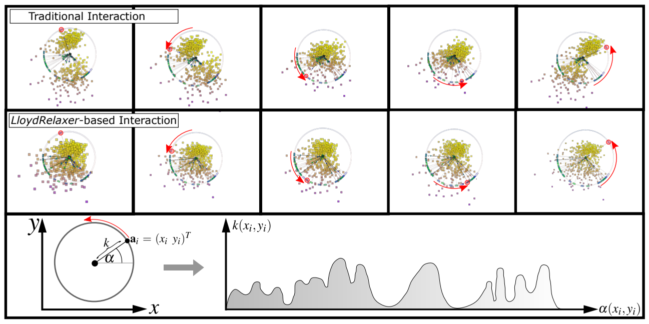

Fig. 13. Traditional Interaction vs. *LloydRelaxer*-based Interaction

In Figure 13 a comparison w.r.t. the WDBC dataset between a traditional interaction (top) and a *LloydRelaxer*-based interaction (middle) is given. Even though the traditional approach indents two clusters, the *LloydRelaxer*-based interaction shows that the clusters cannot be clearly explained by the data. Figure 13 (bottom) shows a typical scaling profile $k(x,y)$ resulting under a *LloydRelaxer*-based interaction, i.e., if the angle α of an anchor point $\mathbf{a} = (x\ y)^T$ is changed (cf. examples in the additional video).

Star Coordinates vs. Radial Visualizations: In [25], a general concept for orthographic, affine, and projective projection has been introduced as

$$\begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{c} & 1 - \sum_{i=1}^{n} \frac{c_i}{n} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{d} \\ 1 \end{pmatrix} \tag{37}$$

where $\mathbf{c} = \mathbf{0}$ gives affine Star Coordinates and $\mathbf{c} = \mathbf{1}$ gives projective Radial Visualizations. By putting (23) into (37) follows

$$\mathbf{p} = \left(\frac{1}{\mathbf{cd} + \sum_{i=1}^{n} \frac{c_i}{n}}\right) \cdot \mathbf{A} \cdot \mathbf{d} = \left(\frac{1}{\mathbf{cd} + \sum_{i=1}^{n} \frac{c_i}{n}}\right) \cdot \overline{\mathbf{A}} \cdot \mathbf{K} \cdot \mathbf{d} \tag{38}$$

meaning that each general projection (e.g., Star Coordinates and Radial Visualizations) comes with an inherent data scaling and thus with the scaling issue as well.

Figure 14 shows that our *LloydRelaxer* can also be applied to Radial Visualizations (and to the complete space of general projections) as is, in order to get the most global structure-enhancing scaling. However, since Radial Visualizations introduce – besides the scaling issues – also non-linear distortions, we focused on Star Coordinates to discuss our *LloydRelaxer*.

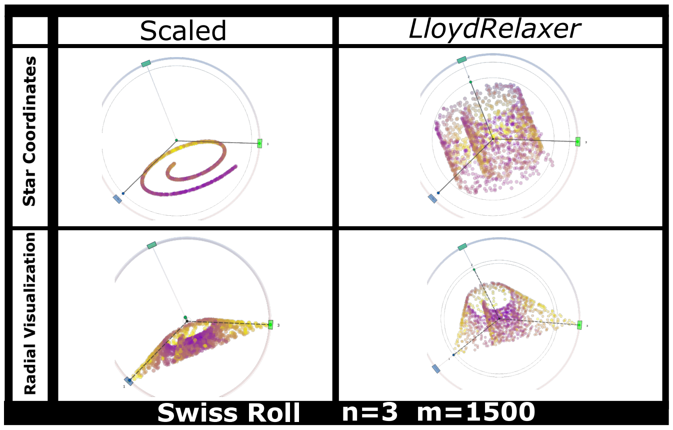

Fig. 14. *LloydRelaxer*: Comparison between Star Coordinates vs. Radial Visualizations for the *Swiss Roll* dataset.

Noise vs. Signal vs. Scaling: A typical data model considers a record as a weighted mixture of a signal and noise component, and as seen in this draft it also shares a scaling property, i.e., informally: data $= \lambda_1 \cdot \text{signal} + \lambda_2 \cdot \text{noise} + \lambda_3 \cdot \text{scaling}$. This is not a complete data model, e.g., the components may be multiplicative instead of additive, they may describe any distributions, may be mutually interlocked, and even more components may be considered; but it is a good approximation for the discussion required at this place. While in this work our *LloydRelaxer* is considered to reduce the influence of the scaling component under projection, the signal and noise components are untouched, meaning that our concept does neither reduce or minimize any noise under projection nor strengthen the signal component.

Choice of axis direction $\overline{\mathbf{A}}$: Working with projection directions raises the question if it is possible to pick up a bad basis of axis directions $\overline{\mathbf{A}}$. The *LloydRelaxer* works and can be used for any axis configuration in $\overline{\mathbf{A}}$, except $\overline{\mathbf{A}} = \mathbf{0}$. If $\overline{\mathbf{A}} = \mathbf{0}$, there is no term $\mathbf{dk} \neq \mathbf{0}$ anymore (cf. Eq. (28)) and the algorithm fails. This is plausible because even $\overline{\mathbf{A}}$ would not encode any direction information anymore on which a scaling could be applied. If just a single column in $\overline{\mathbf{A}}$ is $\mathbf{0}$, e.g., if the user collapses an anchor point to 0, there is still a $\mathbf{dk} > \mathbf{0}$ making the algorithm working as usual. Solely the scaling of the axis which is reflected by this anchor point is not adjusted anymore by the algorithm. In fact, this axis is removed for the scaling analysis process when setting it to 0.

Limitations: First, our approach of the *LloydRelaxer* is solely appropriate under the assumption of having a linear/affine scaling within the data. Non-linear scaling operations (that on the other side do not occur in the Star Coordinates projection process) were not considered and left for future work. Second, our approach is only appropriate for global projections, i.e., local projections, such as LAMP [19] and similar, were not considered. Third, the convergence of the underlying Lloyd relaxation [37] is only guaranteed in low-dimensional spaces (1D or 2D). Thus, our approach is designed for multi-dimensional projections in 2D and it cannot be trivially adopted to the high-dimensional space.

Disadvantages: In addition, there are some shortcomings, e.g., the approach is often only applicable in an interactive state if a data sampling is used. Otherwise, it comes with a large computational overhead. Theoretically, this must not lead to a loss of information, but in practice it can nonetheless, i.e., it might influence the outcome. We scheduled to spend more research effort on these aspects in the future. Moreover, a set of convergence parameters is required and has to be chosen carefully.

Benefits: Our empirical observations suggest that global patterns are worked out well, only dependent on the chosen projection direction in $\overline{\mathbf{A}}$. Thus, the main contribution of our work is to introduce an iterative process in order to mutually interlock the data dimensions to reduce artefacts of an initial (affine) scaling targeting at enhancing the global data structures within the (linear and global) multivariate projection space. This way, one class of standard issues in projection-based data analysis processes can be successfully addressed.

References

[1] G. Albuquerque, M. Eisemann, D. J. Lehmann, H. Theisel, and M. Magnor. Improving the visual analysis of high-dimensional datasets using quality measures. In *Proc. of IEEE VAST*, pages 19–26, 2010.

[2] G. Albuquerque, T. Löwe, and M. Magnor. Synthetic generation of high-dimensional datasets. *IEEE TVCG (Proc. InfoVis)*, 17(12), 2011.

[3] F. Aurenhammer. Voronoi diagrams - a study of a fundamental geometric data structure. *ACM Computing Surveys*, 23:345 – 405, 1991.

[4] E. Bertini. Quality metrics in high-dimensional data visualization: An overview and systematization. *IEEE TVCG*, 17(12):2203–2212, 2011.

[5] H. Chang and D. Yeung. Robust path-based spectral clustering. *Pattern Recognition*, pages 191–203, 2008.

[6] B. N. Delaunay. Sur la sphere vide. *Bulletin of Academy of Sciences of the USSR 7*, (6):793–800, 1934.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[8] D. Dobkin, S. Friedman, and K. Supowit. Delaunay graphs are almost as good as complete graphs. *Disc. and Comp. Geometry*, 5:399–407, 1990.

[9] Q. Du, M. Emelianenko, and L. Ju. Convergence of the lloyd algorithm for computing centroidal voronoi tessellations. *SIAM J. Numer. Anal.*, 44(1):102–119, 2006.

[10] Q. Du, V. Faber, and M. Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM Review*, 41:637–676, 1999.

[11] Q. Du, M. Gunzburger, and L. Ju. Meshfree, probabilistic determination of point sets and support regions for meshfree computing. *Computer Methods in Applied Mechanics in Engineering*, 191:1349–1366, 2002.

[12] R. Etemadpour, R. Motta, J. G. de Souza Paiva, R. Minghim, M. C. F. de Oliveira, and L. Linsen. Perception-based evaluation of projection methods for multidimensional data visualization. *IEEE TVCG (Proc. IEEE InfoVis)*, 21(1):81 – 94, 2015.

[13] M. Fink, J.-H. Haunert, J. Spoerhase, and A. Wolff. Selecting the aspect ratio of a scatter plot based on its delaunay triangulation. *IEEE TVCG (Proc. InfoVis)*, 19(12):2326–2335, 2013.

[14] S. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[15] S. Fortune. Voronoi diagrams and Delaunay triangulations. *Handbook of discrete and computational geometry*, pages 377–388, 1997.

[16] J. H. Friedman. Exploratory projection pursuit. *Jurnal of the American Statistical Association*, (82):249 – 266, 1987.

[17] L. J. Guibas, D. E. Knuth, and M. Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7:381–413, 1992.

[18] B. Joe. Delaunay triangular meshes in convex polygons. *SIAM J. Sci. Stat. Comput.*, 7:514–539, 1986.

[19] P. Joia, D. B. Coimbra, J. A. Cuminato, F. V. Paulovich, and L. G. Nonato. Local affine multidimensional projection. *IEEE Trans. Vis. Comput. Graph.*, 17(12):2563–2571, 2011.

[20] L. Ju, Q. Du, and M. Gunzburger. Probabilistic methods for centroidal voronoi tessellations and their parallel implementations. *Parallel Computing*, 28:1477–1500, 2002.

[21] E. Kandogan. Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. *In Proceedings of the IEEE Information Visualization Symposium*, pages 9–12, 2000.

[22] E. Kandogan. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. *In Proc. of the seventh ACM international Conference on Knowledge Discovery and Data Mining*, pages 107–116, 2001.

[23] D. J. Lehmann, G. Albuquerque, M. Eisemann, M. Magnor, and H. Theisel. Selecting coherent and relevant plots in large scatterplot matrices. *Computer Graphics Forum*, 31(6):1895–1908, 2012.

[24] D. J. Lehmann and H. Theisel. Orthographic star coordinates. *IEEE TVCG*, 19(12):2615–2624, 2013.

[25] D. J. Lehmann and H. Theisel. General projective maps for multidimensional data projection. *Computer Graphics Forum (Proc. Eurographics)*, 35(2):443–453, 2016.

[26] D. J. Lehmann and H. Theisel. Optimal sets of projections of high-dimensional data. *IEEE TVCG (Proc. IEEE InfoVis 2015)*, 22(1):609 – 618, 2016.

[27] M. Lichman. UCI machine learning repository, 2013.

[28] S. Liu, P.-T. Bremer, J. J. Jayaraman, B. Wang, B. Summa, and V. Pascucci. The grassmannian atlas: A general framework for exploring linear projections of high-dimensional data. *Computer Graphics Forum (In Proc. EuroVis)*, 2016.

[29] S. Liu, B. Wang, J. J. Thiagarajan, P.-T. Bremer, and V. Pascucci. Visual Exploration of High-Dimensional Data through Subspace Analysis and Dynamic Projections. *Computer Graphics Forum*, 34(3):271–280, 2015.

[30] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, 2006.

[31] A. Mayorga and M. Gleicher. Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1526–1538, 2013.

[32] K. Nakai and M. Kanehisa. Expert system for predicting protein localization sites in gram-negative bacteria. *Proteins: Structure, Function and Genetics*, 11(2):95–110, 1991.

[33] P. Pagliosa, F. V. Paulovich, R. Minghim, H. Levkowitz, and L. G. Nonato. Projection inspector: Assessment and synthesis of multidimensional projections. *Neurocomputing*, 150, Part B(0):599 – 610, 2015.

[34] F. V. Paulovich, C. T. Silva, and L. G. Nonato. Two-phase mapping for projecting massive data sets. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1281–1290, 2010.

[35] S. Pettie and V. Ramachandran. An optimal minimum spanning tree algorithm. *J. ACM*, 49(1):16–34, 2002.

[36] M. Rubio-Sanchez, F. Diaz, L. Raya, and A. Sanchez. A comparative study between radviz and star coordinates. *IEEE TVCG*, 22(1):619–628, 2016.

[37] M. J. Sabin and R. M. Gray. Global convergence and empirical consistency of the generalized lloyd algorithm. *IEEE Transactions on Information Theory*, 32(2):148–155, 1986.

[38] M. Soltanolkotabi, E. Elhamifar, and E. J. Candès. Robust subspace clustering. *CoRR*, abs/1301.2603, 2013.

[39] J. Stahnke, M. Dörk, B. Muller, and A. Thom. Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions. *IEEE TVCG (Proc. IEEE InfoVis)*, 22(1):629–638, 2016.

[40] W. Street, W. Wolberg, and O. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. *IS&T / SPIE International Symposium on Electronic Imaging: Science and Technology*, 1905:861–870, 1993.

[41] J. Talbot, J. Gerth, and P. Hanrahan. Arc length-based aspect ratio selection. *IEEE TVCG (Proc. InfoVis)*, 2011.

[42] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17:401–419, 1952.

[43] L. Van Der Maaten, E. Postma, and J. Van den Herik. Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10:66–71, 2009.

[44] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[45] S. Waugh. *Extending and benchmarking cascadecorrelation*. PhD thesis, University of Tasmania, 1995.

Dirk J. Lehmann is a senior scientist (Privatdozent) and project manager at the Computer Science Department at the University of Magdeburg, Germany. In 2008, he received a M.Sc. in Computational Visualistics, in 2012 a Ph.D. in Computer Science, and a habilitation (venia legendi) in 2017 from the University of Magdeburg. Since July 2016, he is a visiting professor at the University Rey Juan Carlos in Madrid, Spain. His research interests focus on flow visualization, information visualization and visual analytics.

Holger Theisel is professor for Visual Computing at the Computer Science Department at University of Magdeburg, Germany. In 1994, he received the diploma in Computer Science, in 1996 a Ph.D. in Computer Science, and a habilitation (venia legendi) in 2001 from the University of Rostock. His research interests focus on flow visualization as well as on CAGD, geometry processing and information visualization.