universität wien

DIPLOMARBEIT / DIPLOMA THESIS

Titel der Diplomarbeit / Title of the Diploma Thesis

„ VisCoDeR: A tool for Visually Comparing Dimensionality Reduction Algorithms “

verfasst von / submitted by

René Čutura

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Magister der Naturwissenschaften (Mag.rer.nat.)

Wien, 2018 / Vienna 2018

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 190 884 406
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Lehramtsstudium UF Mathematik UF Informatik und Informatikmanagement
Betreut von / Supervisor:	Ao Univ.Prof. Dr. Renate Motschnig
Mitbetreut von / Co-Supervisor:	Dr. Michael Sedlmair

Contents

This work consists of two parts. The first part is made by me. It deals with the backgrounds of dimensionality reduction techniques itself and their difficulties, the implementation of a learn tool, and evaluation of the tool with some use cases. The second part will be made by Stefan Holzer. In his work, he will give more in-depth considerations about learning itself combined with learning tools, also he will further evaluate the tool with user studies.

Part of the work has been submitted and accepted as a paper to the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2018). Title of the paper is ”VisCoDeR: A Tool for Visually Comparing Dimensionality Reduction Algorithms”. The paper got created together with Stefan Holzer, Michaël Aupetit, and Michael Sedlmair.

1 Abstract

The traditional way of learning dimensionality reduction (DR) techniques is attending courses, or using textbooks, videos, or scientific papers which introduce these DR techniques. All these learning materials have in common, that they are not interactive, and they focus on the mathematical backgrounds and the mechanics of the DR techniques.

With VisCoDeR, I built an interactive online tool, with the purpose to provide interactive material for learning DR techniques. VisCoDeR fosters three modes. The *Discover Mode* allows to qualitatively compare several results of DR techniques by juxtaposing and linking the resulting scatterplots. The *Explore Mode* allows analyzing hundreds of differently parameterized DR results in a quantitative way by directly encode the projections in one scatter plot using t-SNE. The *Play Mode* allows to interactively change parameters on the fly and observe the respective changes in the low-dimensional projections.

Several case studies show that this approach helps to understand similarities and differences between DR techniques, and other important aspects of DR techniques.

2 Zusammenfassung

Üblicherweise lernt man Dimensionsreduktions-Techniken in Kursen, Videos, Büchern oder direkt durch wissenschaftliche Arbeiten welche in die jeweiligen DR-Techniken einführen bzw. diese beschreiben. Diese Lernmaterialien sind meist nicht sehr interaktiv und konzentrieren sich oft auf die mathematischen Hintergründe und die Mechaniken von DR-Techniken.

Mit VisCoDeR habe ich ein interaktives Online Tool erstellt, das als interaktives Lernmaterial verwendet werden kann um den Lernprozess zu unterstützen. VisCoDeR besteht aus drei Modi. Der *Discover Mode* erlaubt das qualitative Vergleichen von unterschiedlichen DR-Techniken, indem die Ergebnisse der DR-Techniken als Scatterplots nebeneinander visualisiert und verknüpft werden. Der *Explore Mode* erlaubt es hunderte Ergebnisse von DR-Techniken quantitativ zu vergleichen, indem die Ergebnisse als Punkte in einem Scatterplot mittels t-SNE visualisiert werden. Der *Play Mode* erlaubt es interaktiv die Parameter während der Iterationen zu ändern um die Auswirkungen am Verhalten beobachten zu können.

Mit Case Studies wird gezeigt, dass dieser Ansatz hilfreich ist um Gemeinsamkeiten und Unterschiede von verschiedenen DR-Techniken zu erkennen, und wichtige Aspekte einzelner DR-Techniken zu verstehen.

3 Introduction

Dimensionality reduction (DR) is an actively researched, growing set of techniques to reduce the dimensionality of data by projecting the data points from the D-dimensional data space to d-dimensional projection space, where $d < D$. DR techniques are used to remove irrelevant dimensions from a dataset, or to visualize high-dimensional datasets. For visualization purposes the dimensionality d of the projection is usually one, two, or three. A dataset can contain thousands of dimensions. For example, if the color values from the pixels of a picture are comprehended as dimensions, then a dataset containing pictures can have millions of dimensions.

DR techniques can be divided into linear and non-linear techniques. Linear DR techniques project high-dimensional data to a lower-dimensional space by mapping the high-dimensional data points linearly to a lower-dimensional space. The advantages of linear DR techniques are the possibility to reverse a projection, and the production of meaningful new dimensions. Non-linear DR techniques project the high-dimensional data non-linearly to a lower-dimensional space. Disadvantages of this non-linear DR techniques are often the meaningless new dimensions these DR techniques produce, but they produce usually better projections than linear DR techniques. Some of the DR techniques have one or more parameters. The users have to set them manually to influence the projections.

When data gets reduced by a DR technique, some information of the data gets inevitably lost during the DR, but other specific information remains. For instance the remaining information could be:

- **Manifolds or structures** with lower intrinsic dimensionality than the dimensionality of the dataset. Manifolds or structures are in a dataset, if the data lies in a lower-dimensional subspace. For example, the known dataset *Swiss roll* has three dimensions, but the data points are lying on a rolled plane like a swiss roll.

- **Similarities or dissimilarities** between the high-dimensional data points. DR techniques use different metrics to measure the similarities from data points. For example, a DR technique can use the euclidean distance to measure the similarity of two data points, or the geodesic distance across the K-nearest neighbor graph in the data space.
- **Outliers or clustered data points** in the high-dimensional dataset. A data point is an outlier if the data point is dissimilar to each other point. If a group of data points is just similar to each data point in this group and dissimilar to each other data point in the dataset, then this group of data points is clustered.

3.1 Motivation

There are two kinds of people dealing with DR: DR-experts which improve existing DR techniques or create new techniques, and DR-naïve users who are new to DR, or use DR techniques to visualize high-dimensional data. In this work I will focus on the second group. In interviews with DR users, Sedlmair et al. outlined seven gaps [SBIM12]. Three of those gaps concern primarily DR-naïve users, who are not familiar with the technical backgrounds of DR techniques. These three gaps are:

- **G1 Conceptual gap** What does DR?
- **G2 Interpretion gap** What do the results mean?
- **G3 Guidance gap** What DR technique or parameters should be used?

To overcome these gaps (G1, G2, G3) without focus on the mathematical backgrounds and mechanics of DR techniques, learning material should help users to develop *mental models* by providing simplified conceptual models.

One problem for DR-naïve users to overcome these gaps is the lack of fitting learning material (besides knowing that the gaps exist). Learning materials for DR techniques are rare, there are a few slides, some courses, textbooks, and of

course some scientific papers introducing DR techniques. These learning materials usually have a heavy focus on the mathematical backgrounds and the mechanics of DR techniques, and beside that they are usually not very interactive.

3.2 Goal

My goal is to develop an interactive tool, that helps users reducing these earlier mentioned gaps (G1, G2, G3). A variety of visualization techniques and interaction techniques should help examining and learning the DR techniques. The tool should serve as an easy-to-understand learning material for the five most researched DR techniques [SZS+17]. These five DR techniques are:

- PCA (principal component analysis) [Pea01]
- MDS (multi dimensional scaling) [Tor52]
- LLE (locally linear embedding) [RS00]
- ISOMAP [TDSL00]
- t-SNE (t-distributed stochastic neighborhood embedding) [MH08]

To create easy-to-understand learning material, I need to consider which data, which visualization- and interaction techniques will help me reach that goal. The research issues are:

- **I1** Can the tool Viscoder developed as part of this thesis help to reduce the gaps G1, G2, and G3 concerning DR-naïve users?
- **I2** Which visualization- or interaction techniques can help reducing the gaps G1, G2, and G3?
- **I3** Will comparative visualization of DR techniques help learn about DR techniques?

The tool will be an online tool implemented via html and java script, so it is platform-independent and can be used by everyone with a browser. I will use d3 [BOH11] for the visualizations, and numeric.js [Loi] for the mathematical operations. I will iteratively design the tool.

4 Theoretical Background

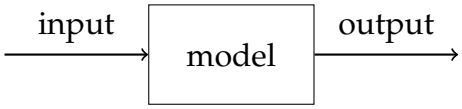

Figure 1: input-output model

Visual parameter space analysis allows users to visually analyze the effects that parameters have to the output of a model with respect to a specific input. In this case, the model is usually the DR technique itself, and the parameters to the model are the parameters to the DR technique. Also, the type of the DR technique can be considered as categorical parameter to the model. Understanding these effects and examining the projections will help reducing the earlier mentioned gaps G1, G2 and G3. Sedlmair et al. [SHB+14] distributed a simple input-output model (see Figure 1), a framework for visual parameter space analysis, and a summarization of parameter space analysis tasks. The framework describes strategies to navigate through parameter spaces. These navigation strategies are:

- **N1 Informed Trial and Error** The user runs a model with a specific setting of input parameters and gets the output of the model shown.
- **N2 Local-to-Global** Like the navigation strategie N1, but with precomputed outputs.
- **N3 Global-to-Local** Starts with an overview over all precomputed outputs. The user can dig down into details.
- **N4 Steering** Parameters can get changed during the computation of the model. Steering is especially used in simulation models.

When using visualization for learning purposes, interaction techniques are essential. Sacha et al. [SZS+17] contributed a categorization of interactions in the DR process model. These interactions got divided in seven scenarios:

- **S1 Data Selection & Emphasis** Filter data which is input for the DR algorithm.
- **S2 Annotation & Labeling** Enriching data records with labels or annotations which affect the DR algorithm's output.
- **S3 Data Manipulation** Manipulate data values to see changes in the output of a DR algorithm.
- **S4 Feature Selection & Emphasis** Change the similarity metrics.
- **S5 Parameter Tuning** Allow to change the parameters of a DR algorithm.
- **S6 Defining Constraints** Set data points to a fixed position to see how the DR deals with these constraints.
- **S7 DR Type Selection** The kind of DR technique itself becomes an input to the model.

DR techniques try to remain different information of the dataset. To reduce the conceptual gap (G1), the user needs to use different DR techniques (S7) and compare their results. To reduce the interpretation gap (G2) the user needs to examine the projections. To reduce the guidance gap (G3) you need to manipulate the parameters of a DR technique (S5) to see the influence from parameters to the projection.

4.1 Multi-dimensional Data

The values of the dimensions from the data have to be numerical and ordinal scaled for DR. If a dataset has more than two dimensions, the dataset can be called multi-dimensional. Nominal scaled dimensions, or rather dimensions containing labels do not count.

4.2 Dimensionality Reduction

There are many DR techniques existing. They project the high-dimensional data considered as $n \times D$ matrix X to a lower-dimensional projection Y considered as $n \times d$ matrix. The projection reduces the high dimensionality D to a lower dimensionality d, where $d < D$. For visualization purposes d is often one, two, or three. To speed up algorithms, where the dimensionality D is a time-limiting factor, the dimensionality d of the projection can have more than three dimensions. These DR techniques can be divided into linear and non-linear techniques. A taxonomy and comparison of DR techniques can be found here [VDMPVdH09]. In the following, I will describe the earlier mentioned DR techniques.

4.2.1 Principal Components Analysis

Principal Components Analysis (PCA) projects the high-dimensional data in a way, that the variance from the high-dimensional data X gets preserved in the projected data Y. PCA is a linear DR technique [Pea01].

Computation The data can be understood as a $n \times D$ matrix X.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

First the covariance matrix C gets computed by centering the data matrix X by subtracting the mean $\overline{x}$ from each data vector x_i. The resulted centered $n \times D$ matrix X_{cent} is used to compute the $D \times D$ covariance matrix C by the dot product

$$C = \frac{1}{n} {X_{cent}}^T \cdot X_{cent}$$

The next step is computing the D eigenvalues and their eigenvectors from the covariance matrix C and selecting the d biggest eigenvalues λ_i and their eigenvectors v_i, where $i \in \{1, \ldots, D\}$). The computed eigenvectors are the principal

components of the data. Out of the d biggest eigenvectors v' a $D \times d$ transformation matrix E gets built.

$$E = \left[v'_1, \ldots, v'_d\right]$$

The projection Y gets computed by the linear transformation $Y = X \cdot E$, which is the reason why PCA is a linear DR technique.

Mental model The data can be considered as point cloud. This point cloud gets lit out of each direction. The projected shadow on the wall can be considered as projection. PCA takes the projection with the biggest extent as result.

4.2.2 Multidimensional Scaling

Multidimensional Scaling (MDS) is a non-linear DR technique. MDS is a set of similar techniques, which differs in the usage of variant similarity metrics. The original DR technique MDS originated from the work of Torgerson. [Tor52]. Metric MDS was introduced by Kruskal [Kru64] in the year 1964. Metric MDS uses a loss- or cost function called *stress*. The dataset X as $n \times D$ matrix gets reduced to Y a $n \times d$ matrix. The stress gets computed based on the distances between x_i to x_j, and y_i to y_j, where $i \neq j \in \{1, \cdots, n\}$ by

$$\text{stress}(X) = \sqrt{\left(\sum_{i \neq j \in \{1,\ldots,n\}} \left[\delta_D(x_i, x_j) - \delta_d(y_i, y_j)\right]^2\right)}$$

Where δ_D describes a metric in the D-dimensional data space and δ_d describes a metric in the d-dimensional projection space.

Computation In the first step all data points y_i ($i \in \{1, \ldots, n\}$) get randomly initialized. Then the stress gets computed and iteratively minimized by an optimization algorithm called *stress majorization*. *SMACOF* [DL05] is such a stress majorization algorithm.

The stress gets minimized in each iteration by creating a ratio matrix Δ

$$\Delta = -\begin{bmatrix} \sum_{i=2}^{n} \frac{\delta_D(x_1,x_i)}{\delta_d(y_1,y_i)} & \frac{\delta_D(x_1,x_2)}{\delta_d(y_1,y_2)} & \cdots & \frac{\delta_D(x_1,x_n)}{\delta_d(y_1,y_n)} \\ \frac{\delta_D(x_2,x_1)}{\delta_d(y_2,y_1)} & \sum_{i=1,i\neq 2}^{n} \frac{\delta_D(x_2,x_i)}{\delta_d(y_2,y_i)} & \cdots & \frac{\delta_D(x_2,x_n)}{\delta_d(y_2,y_n)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta_D(x_n,x_1)}{\delta_d(y_n,y_1)} & \frac{\delta_D(x_n,x_2)}{\delta_d(y_n,y_2)} & \cdots & \sum_{i=1}^{n-1} \frac{\delta_D(x_n,x_i)}{\delta_d(y_n,y_i)} \end{bmatrix}$$

The new $Y^{(i+1)}$ gets set to $Y^{(i+1)} = \frac{1}{n} \cdot \Delta \cdot Y^{(i)}$, where i is the current iteration number. The stopping condition can be a threshold of stress, or a specific number of iterations.

Mental model Consider the data points as point cloud again. Each point gets connected with each other point by a spring without tension. By pressing the point cloud on a plane, the springs get shrunk or stretched. Now the springs restore their original length as good as possible. The point cloud pressed on the plane can be considered as result from MDS. The mental model to MDS is comparable to force-directed layouts [Tut63].

4.2.3 Locally Linear Embedding

Locally Linear Embedding (LLE) is a non-linear DR technique introduced in the year 2000 [RS00]. It tries to remain existing manifolds or structures in the high-dimensional data space. Each data point in the D-dimensional data space can get reconstructed by a linear combination of its k-nearest neighbors. Therefore, the parameter k sets the number of neighbors LLE uses for the k-nearest neighbors. LLE uses the same linear combinations for the data points in the d-dimensional projection space. So, manifolds or structures should remain in the projection, if the intrinsic dimensionality of the manifold comply with d.

Computation In the first step LLE builds a k-nearest neighbor graph by selecting the k-smallest distances from each x_i to each other x_j, where $i, j \in \{1, \ldots, N\}$,

with any k-nearest neighbor algorithm.

In the next step the reconstruction weights in the $n \times n$ weightmatrix W from each data point get computed. Therefore for each point x_i a $k \times D$ matrix Z_i gets created consisting of all neighbors of x_{i_j} ($j \in \{1, \ldots, k\}$) subtracted by x_i.

$$Z_i = \begin{bmatrix} x_{i_1} - x_i \\ \vdots \\ x_{i_k} - x_i \end{bmatrix}$$

The local covariance matrix C_i gets computed by $C_i = {Z_i}^T \cdot Z_i$. The weights w_i corresponding to x_i get computed by solving the linear system

$$C_i \cdot w_i = \mathbb{1} \tag{1}$$

for w_i, where $\mathbb{1}$ is the $D \times 1$ matrix with ones in each entry. To ensure, that the linear system in Equation 1 has a unique solution, the manipulated covariance matrix $C_i' = C_i + \frac{1}{1000} \cdot \mathrm{tr}(C_i)$ is used instead of the covariance matrix C_i. The weight matrix W gets built by setting $W_{ij} = 0$ if j is not a neighbor of i, else W_{ij} is set to the corresponding entry of w_i divided by the sum of all entries of w_i.

Now the projection Y gets computed by using the weights matrix W. A matrix M gets created by $M = (\mathbb{I} - W)^T \cdot (\mathbb{I} - W)$, where $\mathbb{I}$ is the $n \times n$ identity matrix. The smallest $d + 1$ eigenvectors $v_1, \ldots, v_{d+1}$ of M gets computed and the smallest eigenvector v_1 gets discarded. These remaining d eigenvectors are the columns of the $n \times d$ projection matrix Y.

$$Y = \begin{bmatrix} {v_2}^T & \ldots & {v_{d+1}}^T \end{bmatrix}$$

Mental Model LLE tries to unroll manifolds in the high-dimensional data space. For example, if the data points lie on a 2d hilly landscape in a three-dimensional space, LLE tries to flatten the hills.

4.2.4 ISOMAP

ISOMAP was developed in the year 2000 by Joshua B. Tenenbaum, Vin de Slivia, and Jon C. Langford [TDSL00]. ISOMAP is similar to the MDS technique. Instead of using the distances from each point to each other point, a k-nearest neighborhood graph gets built first. Next the geodesic distances (the length of the shortest paths from each point to each other point in the graph) get computed. These geodesic distances get used for the dissimilarity matrix. Therefore, ISOMAP uses the parameter k for defining the number of neighbors each point should consider. The shortest paths can be computed for example with the *Floyd-Warshall algorithm* (algorithm 1), or the *Djisktra's algorithm*. If the neighborhood graph does not fully connect, because a too small k, ISOMAP does not perform good projections.

Computation First, the k-nearest neighborhood graph gets computed. Building the distance matrix Δ by setting $\Delta_{i,j} = \delta_D(x_i, x_j)$, where x_i and x_j are k-nearest neighbors and $\Delta_{i,j} = \infty$ if x_i and x_j are not k-nearest neighbors.

As next step ISOMAP computes the shortest paths out of the distance matrix Δ to get the geodesic distances $\delta_{\text{geod}}(x_i, x_j)$ with a shortest path algorithm.

ISOMAP proceeds like MDS with the geodesic distance matrix Δ_{geod} instead of the distance matrix Δ.

Mental model Like MDS, but each point gets connected only to its k-nearest neighbor. Therefore, after pressing the point cloud to a plane and releasing, the manifold or structure in the high-dimensional data space gets flattened out. Like a crumpled rag gets ironed.

Algorithm 1: Floyd-Warshall algorithm

Input: The distance matrix Δ

Output: The geodesic distance matrix Δ_{geod}

```
for i = 1 to n do
    for j = 1 to n do
        for k = 1 to n do
            /* If the direct distance between x_i and x_j is bigger than
               the distance between x_i and x_j over x_k                  */
            if Δ_{i,j} > Δ_{i,k} + Δ_{k,j} then
                /* Then set Δ_{i,j} to Δ_{i,k} + Δ_{k,j}                  */
                Δ_{i,j} ← Δ_{i,k} + Δ_{k,j};
Δ_geod ← Δ
```

4.2.5 t-distributed Stochastic Neighborhood Embedding

The technique t-SNE was created by Laurens van der Maaten and Geoffrey Hinton to deal with the *crowding problem*. The crowding-problem describes the problem, that in a D-dimensional space it is not possible to lay out more than $D + 1$ points equidistant. This problem appears in projections, when points in the high-dimensional data space are equidistant, and the points get projected to a two-dimensional plane, they can no longer became arranged equidistant. For example in the two-dimensional plane it is not possible to lay out more than three points equidistant [MH08].

Simply spoken, t-SNE adjusts a probability distribution of the projected data points to a probability distribution of the high-dimensional points. The probability distribution is directed by the parameter *perplexity*. T-SNE measures the divergence of the two probability distributions with the *Kullback-Leibler divergence*. This divergence gets minimized by *gradient descent*. Therefore, t-SNE needs the step size ε as parameter.

Algorithm 2: simplified t-SNE

Input: The high-dimensional dataset X and the parameters *perplexity* and ε

Output: The lower-dimensional projection $Y^{(T)}$

begin

search for fitting σ with parameter *perplexity*

compute $p_{j|i} = \frac{exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)}$

compute all $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$

randomly initialize y_i where $i \in \{1, \ldots, n\}$

for *t=1* **to** T **do**

compute all $q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq t} \left(1 + \|y_k - y_t\|^2\right)^{-1}}$

perform gradient descent step with parameter ε to get Y^t

Mental Model The parameter ε is needed for the optimization algorithm gradient descent. When you stand in a hilly landscape, you always go in the direction where your step leads you the deepest. The parameter ε can be interpreted as step length. The parameter *perplexity* can be interpreted as the number of neighbors a point should have.

4.3 Visualization Techniques

Visualizations are interactive graphical illustrations of data. These visualizations should help people solve tasks faster or rather more efficient than without them. [Mun14].

We can only see and think in three dimensions. Classical computer screens can only display two-dimensional images. Visualization techniques help to overcome these problems. It is possible to give the impression of a three-dimensional image on a two-dimensional screen.

More than two dimensions in a scatter plot can be shown by coding the values in a property of the point, such as color, size, shape, or similar. However, the

more information is coded in a point, the less interpretable is the whole scatter plot. For example, a third dimension can be coded in the scatter plot, by sizing the point according to the value of its third dimension (see Figure 2). A rotation of the scatter plot will help see the three-dimensionality of the point cloud. But Sedlmair, Munzner, and Tory [SMT13] suggest to use two-dimensional scatter plots in favor. Two-dimensional scatter plots are often good enough and they come with less interaction costs than three-dimensional scatter plots.

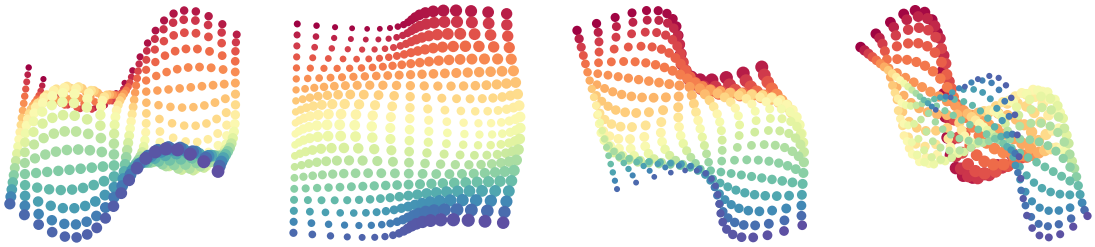

Figure 2: Three-dimensional scatter plot from four perspectives. The sizes of the points correspond with the value of their third dimension.

Some techniques exist for visualizing multi-dimensional data [GTC01].

4.3.1 Scatter Plot Matrix

A scatter plot matrix (SPLOM) shows scatter plots of each combination of dimensions of a dataset (see Figure 3 and Figure 4). With scatter plot matrices come some problems [SMT13]. For example, it is very difficult to see correlations between more than two dimensions. Also, if the dimensionality of the dataset is quite big the scatter plot matrix needs much space.

4.3.2 Parallel Coordinates

Each dimension is a single axis and parallel to each other. A datapoint gets shown as line. The value of the corresponding dimension is the height on the

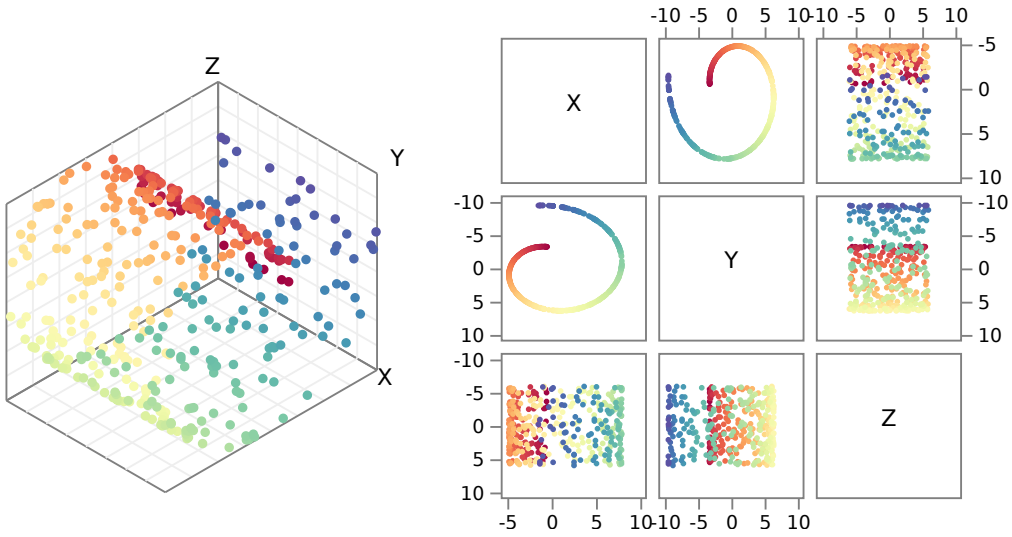

Figure 3: Left three-dimensional scatter plot. Right scatter plot matrix

axis (see Figure 5). Parallel coordinates have the advantage to be very compact, even if the visualized dataset has a higher dimensionality. However, correlations can only be detected between two adjacent axes.

4.3.3 Heatmap

A heatmap is a two-dimensional histogram (see Figure 6-left background). The two-dimensional plane gets divided into areas. For projected data each element in the respective area gets counted, and the area gets appropriately colored. These areas could be rectangular, hexagonal, or similar.

4.3.4 Contourmap

A contourmap (see Figure 6-right background) is similar to a heatmap. A contourmap creates free shapes, for projected data, due to the density of the points in specific areas of the two-dimensional plane. It looks like the contour lines from a map of a landscape, where the points form hills on their position.

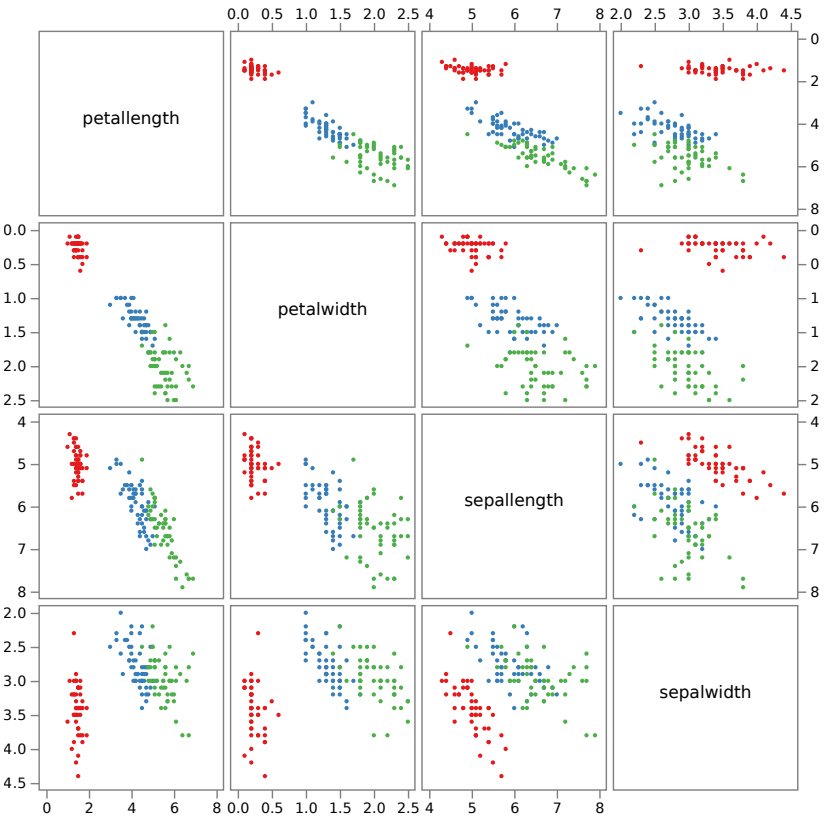

Figure 4: Scatter plot matrix, showing the *Iris* dataset

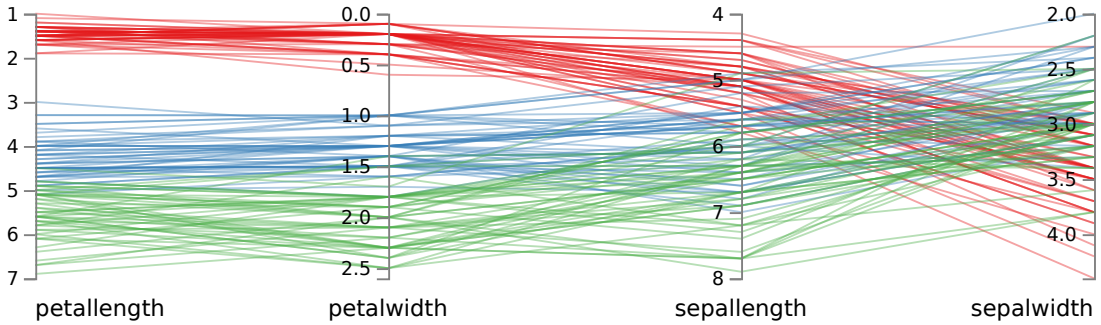

Figure 5: Iris dataset shown in a parallel coordinates plot

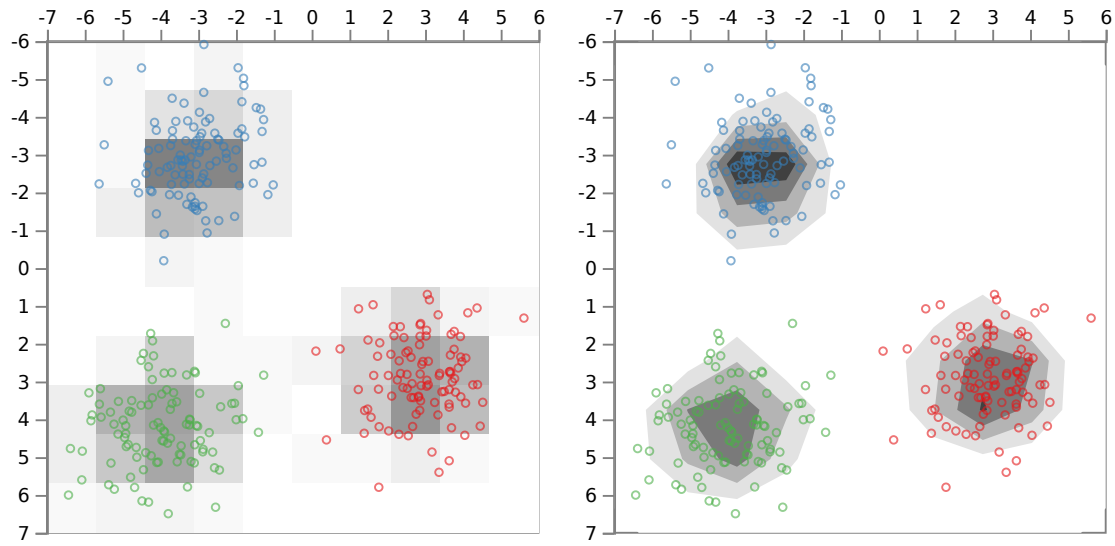

Figure 6: Two scatter plots showing three distinct clusters. In the background on the left side is a heatmap, on the right side is a contourmap

4.4 Component Planes

Component Planes [Ves99] can be used for visualizing the distribution of the dimensions on the projection plane. It has its origin as visualization technique of Self-Organizing Maps. Similar to heatmaps, the projection plane gets divided into areas. These areas can be rectangular, hexagonal, or similar. An area gets color coded with respect to the values of a dimension of the points in this area.

4.5 Proximity Visualization

Proximity Visualization [Aup07], for example used in CheckViz [Aup14] or ProjectionInspector [PPM+15], is a technique to visualize the original neighborhood of a projected point. It helps to identify distortions, which leads to wrong or false neighborhoods. Wrong or false neighborhoods are common errors DR techniques produce [HFA17, MCMT14].

4.6 Related Tools

There are some tools out there, which are related to VisCoDeR. In the following, I will present some related tools. These tools have in common that they focus on using one algorithm at a time. The goal of VisCoDeR goes beyond in that it shows more DR techniques with different parameterization at once. Therefore allowing to compare between different DR techniques. Comparing and picking among different DR techniques and parameterizations is a core challenge of interactive DR [SZS+17].

Probing Projection [SDMT16] is a tool, which lets users examine the projection of a multi-dimensional dataset. It allows to cluster the projected points and examine their details.

How to Use t-SNE effectively [WVJ16] is a website, which has the goal to let users develop intuition for what's going on when projecting data with t-SNE.

Scagexplorer [DW14] shows ways to compare scatter plots, by extracting features and comparing them, like monotonicity, stringiness, or sparseness of the scatter plot. It could be used as metric, which measures a projection of a dataset.

Clustrophile [Dem16] and Clustervision [KEV+17] are similar tools to VisCoDeR, but for cluster algorithms.

ForceSpire [EFN] is a tool that lets users interact with an amount of documents. Therefore it uses DR to visualize the documents in a projection. The main interaction technique is, to change the weights of words in documents to change the projection (S2). There is also further work [EFN12].

Dimstiller [IMI+10] delivers an interactive workflow to guide users visually in selecting features for a DR technique. Their goal is it, to help users understand DR better.

NLDRviz [Ama17] is a similar tool to VisCoDeR. It also visualizes DR projections at the same time at different iteration steps, but offers few interactivity.

5 VisCoDeR

I designed VisCoDeR primarily for students and for learning purposes, secondarily for DR developers, who can compare their DR techniques with other DR techniques and test properties with the inbuilt features. In an iterative design process (see Figure 7) the tool got different modes added. After each step the prototype got discussed in an expert round with two experts: Michael Sedlmair and Michaël Aupetit.

- The *Discover mode* uses juxtaposition to allow for a qualitative side-by-side comparison of different DR results on the same dataset, interactively linked to textual explanations.
- The *Explore mode* allows a direct quantitative comparison between hundreds of differently parameterized DR results displayed altogether in a meta-map using t-SNE.
- The *Play mode* allows testing the robustness of t-SNE, by changing the parameters during its iterations.

At early stages, the *Discover mode* pursued a conventional trial-and-error strategy (N1), inputs to the model were a dataset and the DR type. To offer an overview of the dataset, I simply used a table. At this point, the parameters for the DR techniques could not be set by the user. I focused on showing each output of each iteration of the DR techniques. This helped in no way to reduce the gaps (G1, G2, G3). The performance of this tool was also very bad because I used SVG elements and implemented the DR techniques direct in the website. To solve this problems, I switched from the use of SVG elements to canvas, which burdens not so much the performance of the website. For the DR techniques I used HTML5 Web Workers, which can run in parallel to the website and not burden the user interface.

In the next iterations, the result of the DR techniques got visualized in juxtaposition [GAW+11]. The juxtaposition of projections from different DR techniques

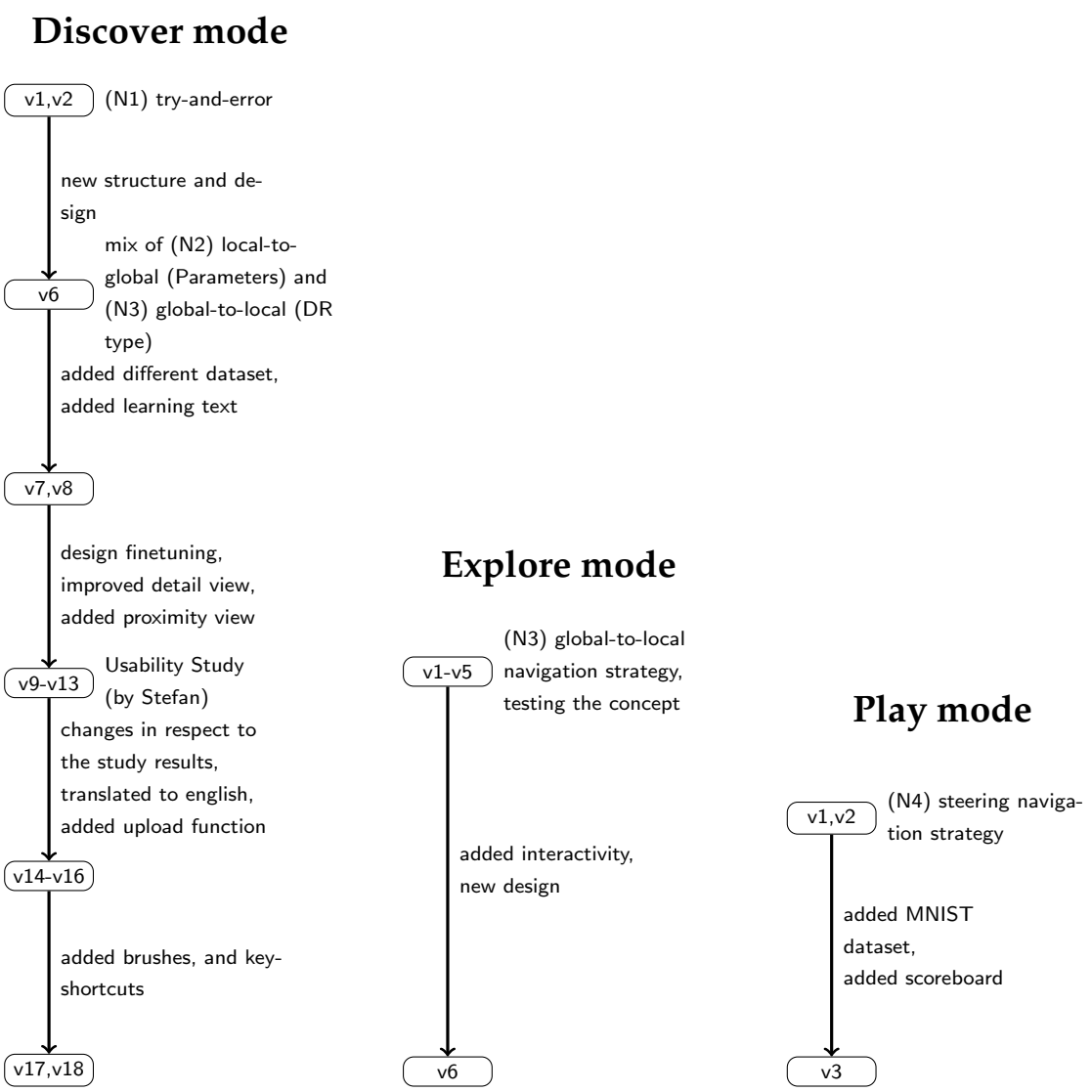

Figure 7: Iterative design process – Timeline

may help reducing the guidance gap (G3), if the learners already know about the DR techniques, because users can examine the effects of the parameters and they can compare the projections. The text (see Figure 8a) functions as guidance and should help to reduce the conceptual gap (G1) with information how the DR techniques work, and explanations what is shown in the visualizations.

In each iteration new features got implemented, the design got improved, and the DR algorithms got refined. The current version of the *Discover mode*[1] (see Figure 8) includes features (described in the next subsections) to help reducing all three gaps (G1, G2, G3).

At some point, I tried the global-to-local navigation strategy (N3) and built the *Explore mode*[2] (see Figure 9). The *Explore mode* shows a visualization of projections from the DR techniques, by projecting the results with t-SNE. The inputs of the model are the results of DR techniques. So, similar projections are shown nearby, dissimilar projections are far from each other.

Another idea was to build the *Play mode*[3] (see Figure 10), which pursues the steering strategy (N4). This navigation strategy is only useful for the iterative DR techniques with parameters (ISOMAP and t-SNE). It allows users to change the parameters during the computation.

[1] `http://renecutura.eu/viscoder/discover.html`
[2] `http://renecutura.eu/viscoder/explore.html`
[3] `http://renecutura.eu/viscoder/play.html`

5.1 Datasets

The datasets consist of name, class, and the dimensions which should get reduced. The class label of each point determines the color of the point in the projection. The colors should help to roughly compare the projections without interaction. The colors have no influence on the DR technique, only the numeric values of the dimensions will be used for projecting the dataset. Three more realistic datasets, five artificial datasets, and the possibility to upload an own dataset in the csv-format are included.

Swiss roll, Waves and S-shape These are artificial datasets, which are easy to understand and to imagine. They are used to show the advantages of LLE and ISOMAP, which are manifold learning DR methods. The *Swiss roll* dataset (see Figure 3) is a well-known dataset, which contains 400 points laid out on a spiral in a three-dimensional space. The Waves dataset contains 400 points, laid out on a hilly landscape. The *S-shape* dataset contains 320 points, similar to the *Swiss roll* dataset but instead of a spiral the points are laid out on a 'S' curve.

Rays and Rays-touching These datasets get generated randomly each time and should be easy-to-understand. You can set the number of dimensions from 3 to 9. It shows the limitations of PCA, the higher the dimensions get the messier the results get. The datasets are similar to the *entangled* dataset Sedlmair et. al [STMT12] used for their taxonomy of visual cluster separation factors.

Spotify These datasets are more realistic than the previous ones. Spotify computes some features from songs, some of them get used in our tool. The dimensions for the DR techniques are

- **Acousticness** is the probability that a song does not use electrical instruments.
- **Dancability** has been computed by some properties of the songs (monotony, bpm, ...). The value of this dimension should say how good the song is to

dance to.
- **Energy** is a measure how intense a song is.
- **Instrumentalness** is the probability that a song is without vocals.
- **Liveness** is the probability, that a song got recorded live.
- **Popularity** is a measure of how likely the song gets played.
- **Valence** is the probability that a song has a negative mood or a positive mood.

Iris The *Iris* dataset [Fis36] consists of fifty examples from the three species of *Iris*. Their four attributes are the length and the width of sepal and petal from the individual example.

MNIST This dataset [LC10] is well-known, and contains handwritten digits from 0 to 9 on a $28px \times 28px$ plane. The grey value of each pixel are acquired as dimensions. I picked only forty of each digit, so the performance of the tool does not suffer that much from the computation of the projections.

5.2 Discover Mode

The *Discover Mode* lets users qualitatively examine results of DR techniques. This is done by juxtaposition the results, interactively link them together, and allowing to examine the projection with various techniques. This mode also offers a learning text with examples to guide users through the tool.

5.2.1 Text

The text contains explanations of the inbuilt DR techniques, examples and links when the focus should be on the visualizations. The text is structured in such a

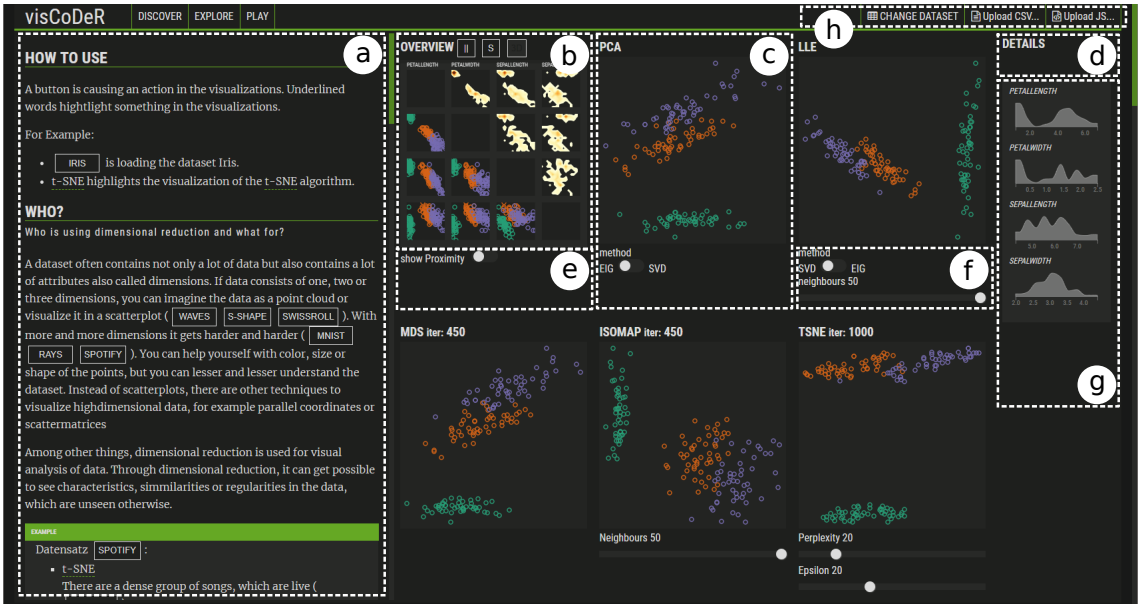

Figure 8: Screenshot of the *Discover mode* . The four sections (a): learning material with explanations and examples, (b): the overview section where users can switch between visualization techniques, (c): DR results where users can highlight data points and see the details in (d) where also the dimensions distribution gets shown as density plot, (e): activate proximity view, (f): parameter settings if the DR technique has any, (g): To show the distribution of the dimensions in the DR scatter plots users can activate them in, (h): users can upload own csv files or DR algorithms with the buttons.

manner that the users begin with a general rough overview and going on digging deeper. There are four sections:

Who and Why? In the first chapter it should get clear, who is using DR and why it is used. The example in this chapter shows — with the *Spotify* dataset — a dense group of songs in the t-SNE result which are all live. The next example shows five songs which separates from all others in the results of LLE, MDS and ISOMAP.

What? The second chapter describes the datasets, *what* is input to the DR technique and what should get visualized.

How? The third chapter describes the five inbuilt DR techniques with mental models. So users can understand what they are doing with the data.

Your turn In the last chapter the users are invited to discover important aspects of DR techniques. For example, if a DR technique is linear or non-linear, or what kinds of distortions exist and how to detect them.

5.2.2 Visualization Techniques

In the *Discover Mode* the high-dimensional datasets get visualized in different ways to give an overview over the selected dataset.

Scatter Plot The projections are visualized as scatter plots. It is the usual way to visualize projected data.

Scatter Plot Matrix (see Figure 8b) On the bottom the areas show scatter plots of the dataset related to the respective dimensions. On the top the areas show density plots of the dataset.

Parallel Coordinates The axes are shown horizontal. This visualization gets usually shown if the dataset has too much dimensions and the scatter plot matrix gets too small.

3D View The data points get shown as a three-dimensional point cloud, which can be rotated in every direction. This view is only possible if the dataset is three-dimensional. It could help imagine the dataset, if the other techniques do not help the user.

Mean Digits The mean digits get shown only for the *MNIST* dataset. It shows the mean values of each pixel of each digit.

Density Plots In the detail section (see Figure 8g) the distribution of values for each dimension gets visualized. The density plots are similar to histograms.

Details If a point gets highlighted, the details of the points get shown in the density plots as vertical lines in the respective color, and the name and class label gets shown above the density plots.

5.2.3 Interaction Techniques

The *Discover mode* allows some interactions with the visualizations.

Sliders The sliders allow users to change the parameters of a DR technique (S5). If a parameter got changed, the computation of the projection restarts with the new parameter settings. For the *Rays* and *Rays-touching* a slider exists to adjust its dimensionality.

Highlighting If a data point gets highlighted in the projection scatter plots, it will get highlighted in each other projection scatter plot and in the overview as well as in the detail view.

Proximity View If activated, it allows to examine the similarities between the data points. Therefore, I use voronoï cells for visualization. Due to the euclidean distance from the highlighted point to each other point, the respective voronoï cell gets colored in scaled gray. The further away, the darker the gray. The closer, the brighter the gray. Also, the twelve nearest points get connected with a white line.

Dimensional Distribution The density plots in the detail view are selectable. If a density plot gets selected, the respective dimensional distribution gets shown on the projection scatter plots as heatmap in the background. I used the same technique as in probing projection [SDMT16]. The projection plane is divided in 8×8 areas. Each area is colored with a gray scaled value, based on the mean of the values of the respective dimension from the points in this area. If an area is empty, the mean of the values of the respective dimension from the three closest points is used.

Brush The brush allows to highlight multiple points in all scatter plots, by selecting them in one scatter plot.

5.3 Explore Mode

The *Explore Mode* shows 1004 (1 PCA, 1 MDS, 45 LLE, 45 ISOMAP, 912 t-SNE) precomputed DR projections at once. These DR projections considered as datapoints of a meta dataset need to get normalized and preprocessed in order to function as input to a DR technique. It is important to preprocess the meta dataset because the projections of DR techniques are often similar to each other, but rotated or mirrored. This could lead to unwanted behaviors of the meta DR technique. For example: it could happen, that one axis of the meta projection reflects the angle of rotation.

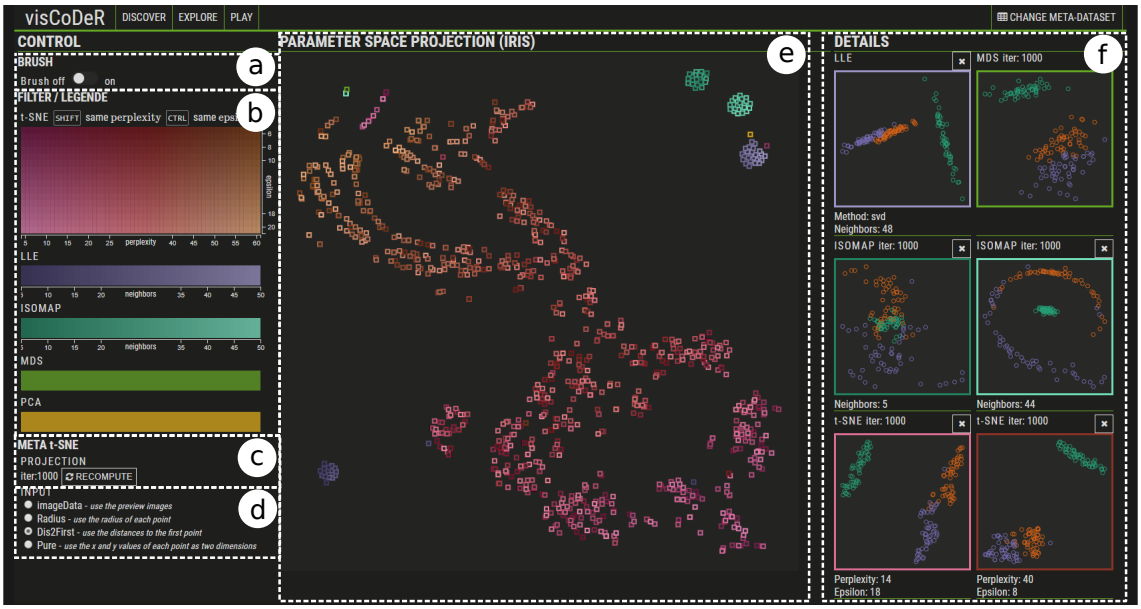

Figure 9: Screenshot of the *Explore mode*. The control section on the left contains (a): a switch to turn on brushing, (b): the filters where users can select results with specific parameters, (c): a button to recompute the visualization with the method users can set in (d). The main view (e), where all results are shown as circles and if users hover over one they expand to a preview or when users click on them they will see the result in (f) the detail section.

5.3.1 Datasets

Some of the datasets of the *Discover Mode* get used in the *Explore Mode*, the *MNIST*, *Iris*, and the *Swiss roll* dataset. To remove the rotations, In earlier versions of the prototype (Figure 7-Explore mode v1-v5) I used an optimizer from numeric.js to minimize the sum Δ of the distances δ of the points y_i of a fixed projection Y to the points $\widetilde{x}_i$ of a projection $\widetilde{X}$ by rotating the projection X at an angle of α, where $i \in (1, \ldots, N)$ with N as the number of points in the projection.

$$\min_{\alpha}\Big(\Delta\left(\alpha\right)\Big), \text{ where}$$

$$\Delta(\alpha) = \sum_{i=1}^{N} \delta(x_{\alpha i}, y_i), \text{ where}$$

$$X_\alpha = R(X, \alpha) = X \cdot \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}.$$

This approach did not work out that well, because it does not remove the reflections. An all-in-one approach is to use rigid transformation to rotate and mirror the projections as needed. The covariance matrix $C = Y^T \cdot X$ needs to get factorized per singular-value decomposition $C = U\Sigma V^T$. The transformation matrix $R = U^T \cdot V$ rotates and mirrors the projection $\widetilde{X}$ in such a way, that the distances of each point of $\widetilde{X}$ to the points of the fixed projection Y are minimal, by the dot-product $\widetilde{X} = X \cdot R$.

The preprocessed meta datasets are still not ready to function as input for the meta DR technique. A point of the meta dataset has N dimensions. N is the number of points in the unprojected dataset. Each Dimension has two values, the x and y position of the projected point. To transform a meta data point to a usable vector, a user can select in the *Explore Mode* out of four options:

- **imgData**: A small image of the respective projection scatter plot gets used as input. Each pixel's gray value complies with the number of points on it.
- **radius**: The radii of each point to the origin $(0, 0)$ are used as input.

- **dis2first**: The distances of each point to the first point in the projection get used as input.
- **pure**: The x and y values of the points are used as input.

After the selection of the input option, the meta t-SNE can be applied on the (preprocessed) results of the DR techniques. By doing so, similar results are visualized nearby, dissimilar results are visualized far apart.

5.3.2 Visualization Techniques

Filter and Legend The projections get placed on a grid and color coded due to their parameters and DR type (see Figure 9a). As the DR technique with two parameters t-SNE has a 2d color scale, the x-axis for perplexity is hue scaled and on the y-axis (ε) the lumination varies. ISOMAP and LLE have a single parameter (k...neighbors), which is coded as lumination on the x-axis. PCA and MDS have no parameter, therefore they do not need a scale.

Parameter Space Projection In this section (see Figure 9e) the meta projection is shown as scatter plot. Each point on the scatter plot complies with a result of a set consisting of a DR technique and associated parameters. By hovering over a point, the point expands to a preview of the scatter plot, which is also used as input for the meta t-SNE if "imgData" is selected. This allows the user, for example, to examine the projections of the different DR techniques with respect to the sensitivity of the parameters.

Detail View The section (see Figure 9f) is empty as long as no point has been pressed. If a point gets pressed in the parameter space projection, a bigger preview of the respective projection gets shown. By clicking on the preview in the detail view, the projection gets recomputed.

5.3.3 Interaction Techniques

The *Explore mode* needs also some abilities to interact with the visualizations.

Highlighting By hovering over a point in the parameter space projection or on an area in the filter the respective area or point get highlighted. By clicking on a point or an area, the set of DR type and associated parameters get shown in the detail view.

Brush The brush can be activated in the control section (see Figure 9b). It allows to brush in the parameter space projection. The selection will get highlighted in the parameter space projection and in the filter/legend. This interaction technique allows to examine the sensitivity of the DR techniques.

Recompute The projection in the parameter space projection is precomputed. If an other input gets selected (in Figure 9d), then the meta t-SNE computes it automatically. However, if a user wants to compute the projection again, then there is a button for it (see Figure 9c)

5.4 Play Mode

The steering strategy (N4) is just useful for t-SNE and ISOMAP, because they are the only implemented DR techniques which are iterative and have parameters. The optimization works without stopping condition. If a parameter is changed, the DR technique continues with the changed respective parameter.

Shaker With the shaker button, each point in the projection gets a little bit shifted. So users can examine the robustness or stability of the DR technique. After shaking, the projection gets registered in the scoreboard (see Figure 10-right) with some additional information, for instance, the parameterization and the actual iteration step.

Figure 10: Screenshot of the *Play mode*. (left): The Playground shows the result of a t-SNE iteration step. (middle): The control section, where users can ”shake” the result, change the parameterization, and see the actual cost of the projection. (right): A new score gets inserted by shaking the projection. The scores in the scoreboard contain the results that got ”shaked”, their parameterization, the iteration step, and the cost of the projection.

6 Implementation

For implementation, I used d3 [BOH11] for visualization, numeric.js [Loi] for linear algebra tasks and HTML5 Web Workers [web] to run the DR algorithms in parallel. I implemented PCA, MDS, ISOMAP and LLE by myself, and used a ready-to-use implementation for t-SNE, but I had to do minor changes to fit it in my tools.

Initially, I had serious problems with the performance of the DR algorithms, because I implemented them without knowing about HTML5 Web Worker. This HTML5 Web Worker allows to run a script parallel to the main script and deliver results with callbacks. By running parallel to the main script, the DR techniques did not influence the performance of the UI anymore. This allowed me to show the visualizations for the DR techniques in juxtaposition with reasonable performance.

I began to visualize the scatter plots in SVG elements. Due to the number of points the usage of SVG elements turned out to be not performant enough. So I switched to canvas, which is a bit harder to let user interact with the visualizations, because canvas only shows images.

7 Evaluation

In this section, I will evaluate VisCoDeR with user scenarios. Further evaluations, like user studies, will be made by Stefan Holzer.

7.1 Discover Mode

The main concept of the learning mode, is the juxtaposition of the projections of different DR techniques. The textual description functions as verbal explanation of the DR techniques. In the traditional way to learn DR techniques, learners have to learn principles of DR techniques in class, youtube, or books. Then they need to call DR techniques with R, python or similar, to create a scatter plot. The learner needs to remember the scatter plot, create a new one with different parameterization and compare the results, and so on. The learning mode of VisCoDeR combines all prerequisites.

7.1.1 Use Case 1: Limitations of DR algorithms

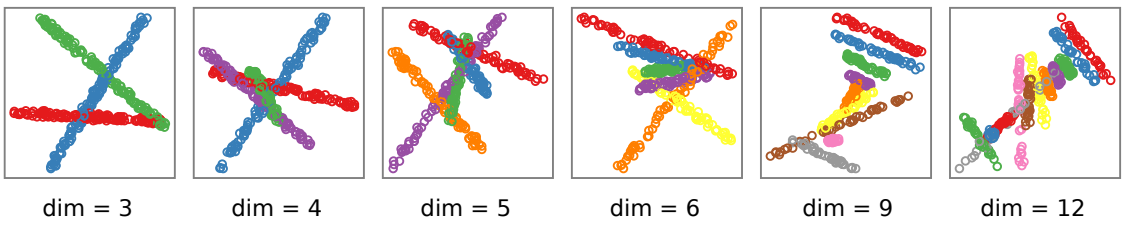

Figure 11: Rays with variant dimensionality get projected to a two-dimensional projection space with PCA. Each ray is distinctly colored.

It is hard to gain intuition about the limitations of DR techniques, when you are learning them from a book. For example: PCA is failing if the dimensionality of the dataset is too high (see Figure 11). The *Rays* dataset allows users to increase and decrease its dimensionality. This dataset gets generated each time. For each dimension a narrow beam of points is created which expands in the direction of

the respective dimension. If users increase the dimensionality too much, they see PCA failing. At a certain number of dimensions the PCA result shows only a blob of points. It happens sometime in MDS, ISOMAP, and t-SNE results, that a ray gets crossed by another (see Figure 12). Crossed rays should not exist, because the rays do not touch in the high-dimensional space.

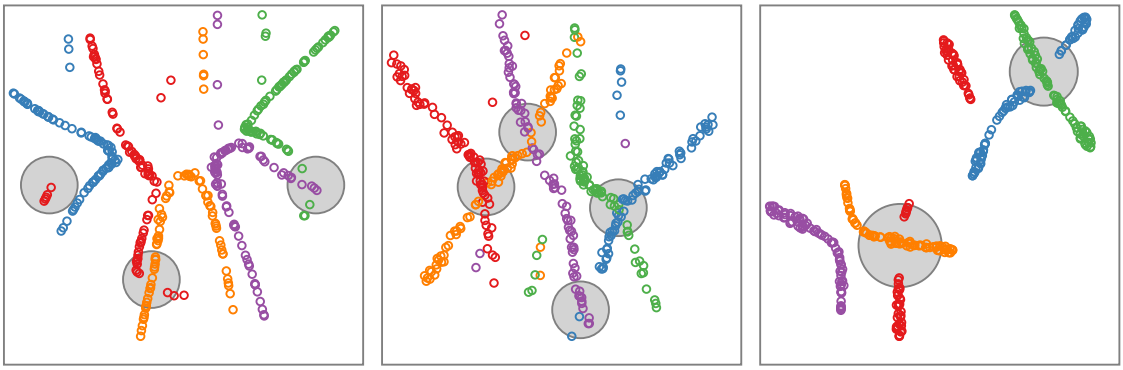

Figure 12: Rays dataset with five dimensions, projected with ISOMAP, MDS, and t-SNE. Each ray is distinctly colored.

7.1.2 Use Case 2: Choose a DR algorithm

For the *MNIST* dataset t-SNE generates the best results (see Figure 13). The digits are clearly separated from other digits, and if a digit is in the wrong cluster, the reasons are kind of comprehensible. PCA delivers no result in this prototype because of limitations of the browser. With high-dimensional datasets — like the *MNIST* dataset — you have to deal with the curse of dimensionality. This problem occurs, because the high-dimensional euclidean distance makes less sense in such high dimensions, therefore high-dimensions points become more and more similar. MDS, ISOMAP, and LLE use euclidean distances, t-SNE uses probabilities that are not that responsive to too much dimensions.

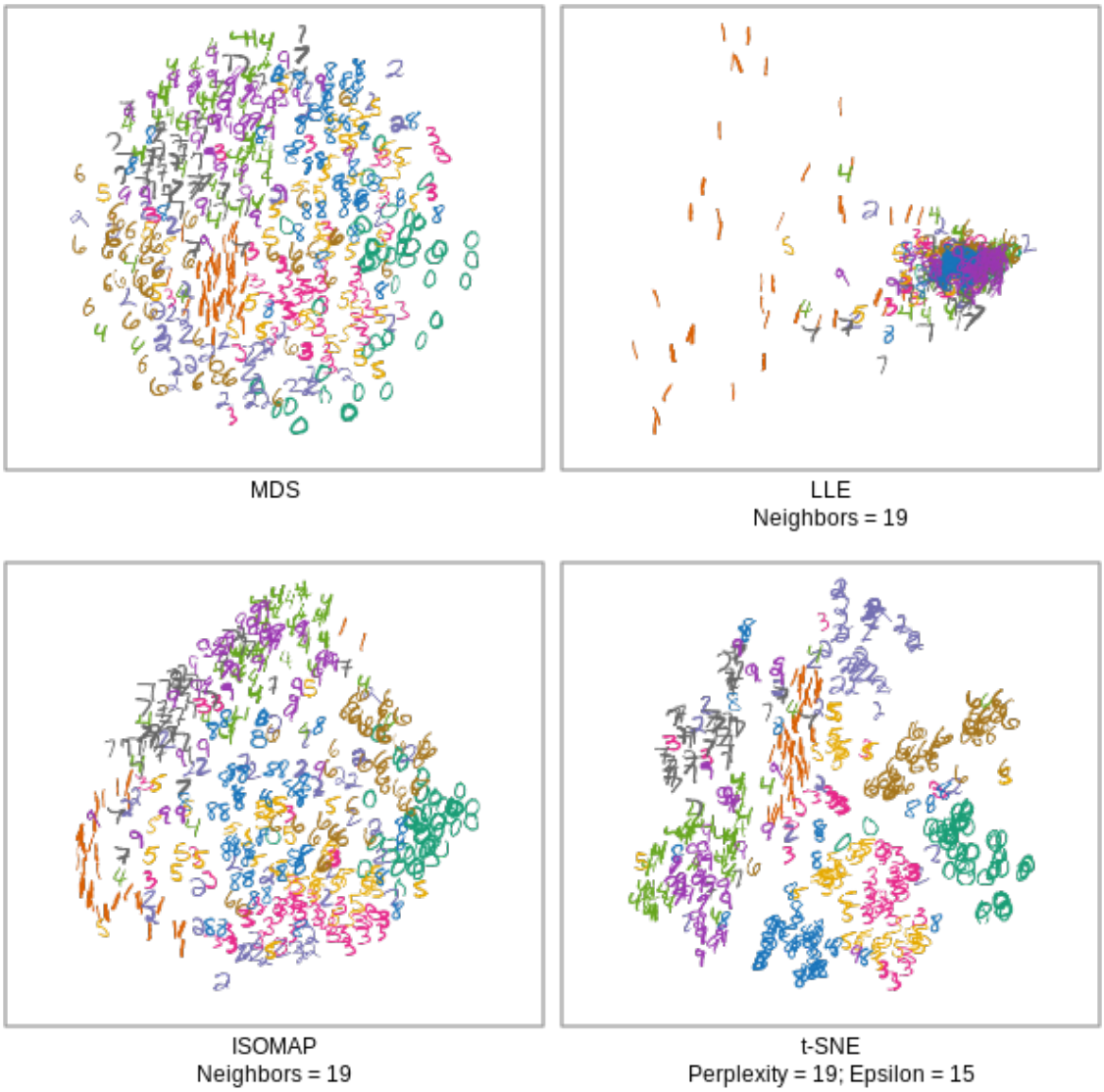

Figure 13: Projections of the *MNIST* dataset. The typical circle shape of MDS comes from the curse of dimensionality. This problem occurs because of the high dimensionality of the *MNIST* dataset. LLE has a bad output in this case. ISOMAP with number of neighbors $K = 19$ has better results than MDS, because the digits are better separated. t-SNE generated the best results, the digits are good separated, with few exceptions.

7.2 Explore Mode

The *Discover Mode* is good for viewing and interactively comparing a few DR algorithms. However, it does not scale. For instance, what if we want to more thoroughly investigate different parameterizations of algorithms. For example, t-SNE leads to large variations given different parameterizations.

The main concept to deal with that problems is visual parameter space analysis [SHB+14]. Create many samples, compute a similarity measure, and show the samples as an overview. This overview allows users to drill down. Sedlmair et al. [SHB+14] outlined some analysis tasks concerning visual parameter space analysis, which are partially supported within this mode. These tasks are:

- **T1 Optimization** *"Find the best parameter combination given some objectives."*
- **T2 Partitioning** *"How many different types of model behaviors are possible?"*
- **T3 Fitting** *"Where in the input parameter space would actual measured data occur?"*
- **T4 Outliers** *"What outputs are special?"*
- **T5 Uncertainty** *"How reliable is the output?"*
- **T6 Sensitivity** *"What ranges/variations of outputs to expect with changes of input?"*

In the following, I will discuss use cases for (T6) Sensitivity, (T2) Partitioning, (T1) Optimization.

7.2.1 Use Case 1: Sensitivity Analysis

Sensitivity says how changes to the parameterization influence the results. By using the meta t-SNE similar results are nearby. To see the parameterization, users can hover over the areas in the colored parameter space and highlight the

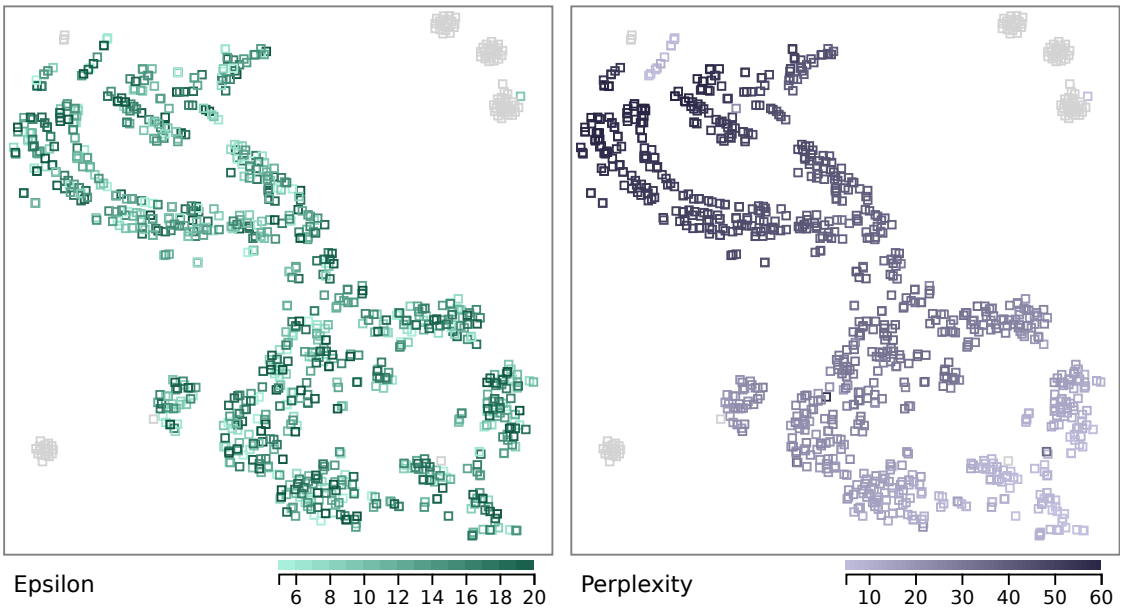

Figure 14: The meta projection of the *Iris* dataset. The colored points represent results of t-SNE with various parameterization. (left): The darker the point, the higher the value of ε, the brighter the point, the lower the values of ε. (right): The darker the point, the higher the value of perplexity, the brighter the point, the lower the values of perplexity.

respective results in the meta t-SNE projection. Because t-SNE has two parameters it is possible by pressing `SHIFT` or `CTRL` to highlight every result with the same respective parameter. This shows users that the perplexity parameter is more sensitive than the ε parameter. Because the results with the same perplexity and different ε are close to each other (see Figure 14-right), the results with the same ε and different perplexity are scattered over the whole parameter space projection (see Figure 14-left). This leads to the conclusion that the perplexity parameter is very sensitive to changes and ε has no big effects on the result.

7.2.2 Use Case 2: Partitioning

The main question of partitioning is how many behaviors of a DR technique can be detected. For instance, the *Iris* dataset consists of two clusters. If the

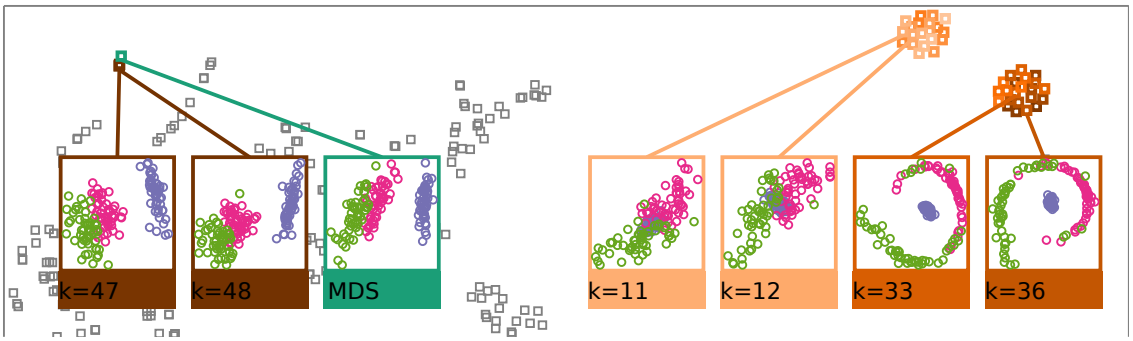

Figure 15: A part of the meta t-SNE visualization of results with the Iris dataset. Left Cluster of ISOMAP with value k as parameter neighbor is near the MDS result. Two clusters on the right with bad parameterization, which leads to two distinct behaviors.

parameter for neighbors in ISOMAP is too low, the neighborhood graph does not connect these two clusters. The not connected neighborhood graph leads to bad results, which are clustered in the meta t-SNE projection (see Figure 15 $k = 11$, $k = 12$). If the parameter is high enough the neighborhood graph connects, but not good enough for good results (see Figure 15 $k = 33$, $k = 36$). The higher the parameter gets, the more similar the results are to the MDS result. The third cluster of ISOMAP results is nearby the MDS result (see Figure 15 $k = 47$, $k = 48$, and MDS).

7.2.3 Use Case 3: Optimization

A common task with visual parameter space analysis [SHB+14] is to find a result which fits best to a user's need. In this case, the goal of the optimization task is to find the best fitting combination of DR type and parameterization. By clicking on the results, users can recompute the DR algorithm with the respective parameterization. The users can compare the results and pick the best one.

The task in the partition use case can also be considered as an optimization task, where a user has to find the best fitting ISOMAP projection. The user could find the solution by hovering over the filter, and inspect promising projections in

detail until a projection fits. Also, the user could browse through the parameter space projection, if the snippet images are not fitting, then the projections nearby would fit neither.

7.3 Play Mode

The *Play Mode* allows users to change the parameterization of t-SNE during its computation. This is the navigation strategy steering (N4).

7.3.1 Use Case 1: Stability

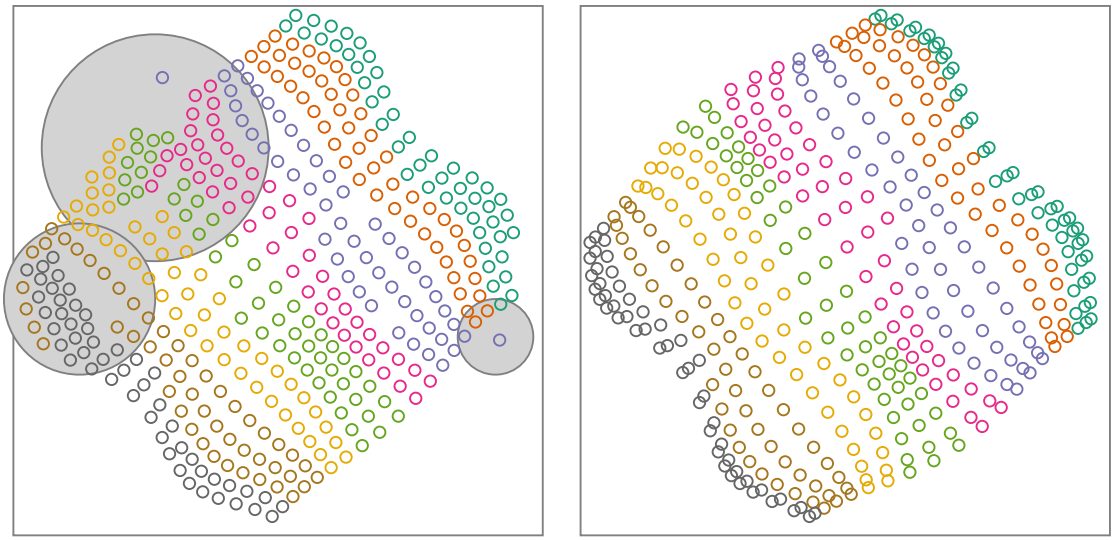

Figure 16: *Waves* dataset (left): optimizer got stuck on local minimum, leads to bad projection with groups of points in the wrong place (circles). (right): after shaking the optimizer escaped the local minimum.

The *Play Mode* allows users to test t-SNE on its stability. It is important for users to gain trust in a DR technique. This trust can grow, if a DR technique is stable. Stability of a DR technique means, that the results are the same under variant initial conditions. As mentioned earlier, the first step of t-SNE is, to initialize the projection with randomly placed points. The cost function of t-SNE defines the ”badness” of the projection, which gets minimized. For the most optimization techniques it is possible, that the minimizer get stucked in a local minimum. The projection get stucked on a local minimum, if the projection converges to a

bad result (see Figure 16-left). A local minimum should be avoided, because the projection could be better, but the minimizer did not find it.

By pressing the shake button (see Figure 10) the points are shaked by adding random noise to their positions. It is possible to escape a local minimum. This can be seen if the projection converged to a bad result (see Figure 16-left) and after shaking the projection changes to a good result (see Figure 16-right). This can be easily detected on an easy-to-understand dataset like the *Waves* dataset.

8 Conclusions and future work

In the traditional way to learn DR techniques, learners have to learn principles of DR techniques in class, youtube, or books, to learn about the basics and mathematical backgrounds. Then learners need experience with DR techniques to dig deeper. Therefore, they need to utilize R, python or similar to create static scatter plots, and remember them to compare them with scatter plots of other DR techniques. By using VisCoDeR, which visualizes scatter plots of more than one DR technique, the user gets supported. On the one hand, the user does not need to utilize R, python or similar, on the other hand more than one DR technique gets shown. With VisCoDeR the user has a powerful ready-to-use tool to learn about DR techniques.

The text section in the *Discover Mode* can help reducing the conceptual gap (G1). It describes the fundamentals of DR techniques supported by visualizations. With variant interaction tools the learners can examine what the results mean, so the interpretion gap (G2) can be reduced. By examining the parameter space with the *Explore Mode* the users can learn about the differences between DR algorithms and their behaviors with respect to the parameterization. Therefore VisCoDeR can help reducing the guidance gap (G3). In the end, VisCoDeR can help reducing these gaps (I1).

With the ability to examine the scatter plots with variant interaction techniques, the learning process of the users can get supported. By examining the DR results with the proximity visualization, the users are able to better understand what the respective DR technique are doing to data (G1). The component planes show the distribution of the dimensions on the projection and help gain intuition about the projecting dimensions (G2). Brushing and Linking in the *Discover Mode* and the direct encoding of hundreds of DR results in the *Explore Mode* enables the user to compare the results of different DR techniques (G3). With this visualization and interaction techniques it is possible to help learners reduce the gaps (I2).

The *Discover Mode* visualizes different DR techniques in juxtaposition. This helps users to compare between different DR techniques qualitatively. The *Explore Mode* shows the DR techniques with hundreds of different parameterizations, which allows users to compare them quantitatively (I3).

8.1 Future Work

There is plenty of possible future work, for example:

The use case study alone is not enough to tell to which extent VisCoDeR helps reducing the gaps (G1, G2, G3). A user study would help evaluate VisCoDeR in more detail.

The learning text could be expanded to the *Explore mode* and the *Play mode*. Therefore the different modes could be merged to one big tool. This allows a potential better learning flow for the users. The tool also could be rebuilt with another narrative flow. Narrative flows got reviewed [MHRL+17], showing different kinds of narrative flows and their advantages.

It should be possible to add more interactivity to the tool, by allowing changes in the data (S1, S2, S3, S6). This could help reducing other gaps than the previously mentioned ones (G1, G2, G3).

Like the *Explore mode* visualizes the projections of different sets of DR technique and parameterization, the individual iteration steps could also be visualized.

I have implemented the five most researched DR techniques and built in a possibility to upload own or other DR techniques. However, there could be more pre-implemented DR techniques to select, for example an autoencoder, or improvements of LLE.

People who use data analysis techniques could encounter similar problems as people who use DR. Therefore, it could be possible to built similar tools for other data analysis techniques. For example, ClusterVision [KEV+17] is a similar tool to VisCoDeR, but for clustering algorithms.

Clustrophile [Dem16] has very cool features, forward-projection, backward-projection and prolines. However, to add these features I need out-of-sample extensions for the DR techniques. By implementing them, they could be very useful for closing the interpretation gap (G2), because the projections could be examined directly. But, because of the nonlinear nature of the most inbuilt DR techniques, it is very difficult to find out-of-sample extensions.

References

[Ama17] Lorenzo Amabili. Visualizing algorithms of nonlinear dimensionality reduction techniques. 2017.

[Aup07] Michaël Aupetit. Visualizing distortions and recovering topology in continuous projection techniques. *Neurocomputing*, 70(7):1304–1330, 2007.

[Aup14] Michaël Aupetit. Sanity check for class-coloring-based evaluation of dimension reduction techniques. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 134–141. ACM, 2014.

[BOH11] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[Dem16] Çagatay Demiralp. Clustrophile: A tool for visual clustering analysis. 2016.

[DL05] Jan De Leeuw. Applications of convex analysis to multidimensional scaling. *Department of Statistics, UCLA*, 2005.

[DW14] Tuan Nhon Dang and Leland Wilkinson. Scagexplorer: Exploring scatterplots by their scagnostics. In *Visualization Symposium (PacificVis), 2014 IEEE Pacific*, pages 73–80. IEEE, 2014.

[EFN] Alex Endert, Patrick Fiaux, and Chris North. Forcespire: Semantic interaction for visual analytics.

[EFN12] Alex Endert, Patrick Fiaux, and Chris North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 473–482. ACM, 2012.

[Fis36] Fisher. Iris dataset. `https://archive.ics.uci.edu/ml/datasets/iris`, 1936.

[GAW+11] Michael Gleicher, Danielle Albers, Rick Walker, Ilir Jusufi, Charles D Hansen, and Jonathan C Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011.

[GTC01] Georges Grinstein, Marjan Trutschl, and Urška Cvek. High-dimensional visualizations. In *Proceedings of the Visual Data Mining Workshop, KDD*, 2001.

[HFA17] Nicolas Heulot, Jean-Daniel Fekete, and Michael Aupetit. Visualizing dimensionality reduction artifacts: An evaluation. *arXiv preprint arXiv:1705.05283*, 2017.

[IMI+10] Stephen Ingram, Tamara Munzner, Veronika Irvine, Melanie Tory, Steven Bergner, and Torsten Möller. Dimstiller: Workflows for dimensional analysis and reduction. In *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, pages 3–10. IEEE, 2010.

[KEV+17] Bum Chul Kwon, Ben Eysenbach, Janu Verma, Kenney Ng, Walter F Stewart, Adam Perer, et al. Clustervision: Visual supervision of unsupervised clustering. *IEEE Transactions on Visualization and Computer Graphics*, 2017.

[Kru64] Joseph B Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[LC10] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. `http://yann.lecun.com/exdb/mnist/`, 2010.

[Loi] Sébastien Loisel. Numeric javascript. `http://www.numericjs.com/`.

[MCMT14] Rafael Messias Martins, Danilo Barbosa Coimbra, Rosane Minghim, and Alexandru C Telea. Visual analysis of dimensionality reduction quality for parameterized projections. *Computers & Graphics*, 41:26–42, 2014.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[MHRL+17] S McKenna, N Henry Riche, B Lee, J Boy, and M Meyer. Visual narrative flow: Exploring factors shaping data visualization story reading experiences. In *Computer Graphics Forum*, volume 36, pages 377–387. Wiley Online Library, 2017.

[Mun14] Tamara Munzner. *Visualization analysis and design*. CRC press, 2014.

[Pea01] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[PPM+15] Paulo Pagliosa, Fernando V Paulovich, Rosane Minghim, Haim Levkowitz, and Luis Gustavo Nonato. Projection inspector: Assessment and synthesis of multidimensional projections. *Neurocomputing*, 150:599–610, 2015.

[RS00] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[SBIM12] M Sedlmair, M Brehmer, S Ingram, and T Munzner. Dimensionality reduction in the wild: Gaps and guidance. *Dept. Comput. Sci., Univ. British Columbia, Vancouver, BC, Canada, Tech. Rep. TR-2012-03*, 2012.

[SDMT16] Julian Stahnke, Marian Dörk, Boris Müller, and Andreas Thom. Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions. *IEEE transactions on visualization and computer graphics*, 22(1):629–638, 2016.

[SHB+14] Michael Sedlmair, Christoph Heinzl, Stefan Bruckner, Harald Piringer, and Torsten Möller. Visual parameter space analysis: A conceptual framework. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2161–2170, 2014.

[SMT13] Michael Sedlmair, Tamara Munzner, and Melanie Tory. Empirical guidance on scatterplot and dimension reduction technique choices. *IEEE transactions on visualization and computer graphics*, 19(12):2634–2643, 2013.

[STMT12] Michael Sedlmair, Andrada Tatu, Tamara Munzner, and Melanie Tory. A taxonomy of visual cluster separation factors. *Computer Graphics Forum (Proc. EuroVis)*, 31(3):1335–1344, 2012.

[SZS+17] Dominik Sacha, Leishi Zhang, Michael Sedlmair, John A Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C North, and Daniel A Keim. Visual interaction with dimensionality reduction: A structured literature analysis. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):241–250, 2017.

[TDSL00] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[Tor52] Warren S Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, 1952.

[Tut63] William Thomas Tutte. How to draw a graph. *Proceedings of the London Mathematical Society*, 3(1):743–767, 1963.

[VDMPVdH09] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10:66–71, 2009.

[Ves99] Juha Vesanto. Som-based data visualization methods. *Intelligent data analysis*, 3(2):111–126, 1999.

[web] `https://www.w3.org/TR/workers/`.

[WVJ16] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-SNE effectively. *Distill*, 2016.