5 Visualizing the Output Space

Projection methods are a common approach to dimensionality reduction with the aim of transforming high-dimensional data into a low-dimensional space. For data visualization purposes, projections into two dimensions are considered here. However, when the output space is limited to two dimensions, the low-dimensional similarities cannot completely represent the high-dimensional distances, which can result in a misleading interpretation of the underlying structures.
Nonetheless, visualization techniques based on scatter plots produced using a projection method (usually principal component analysis (PCA)) remain the state of the art in cluster analysis (e.g., [Everitt et al., 2001, pp. 31-32; Hennig et al., 2015, pp. 119-120, 683-684; Mirkin, 2005, p. 25; G. Ritter, 2014, p. 223]). Even if one disregards that "PCA remains a rather basic method and suffers from many shortcomings" [Lee/Verleysen, 2007, p. 226], visualization based on such a scatter plot is questionable in principle. Several two-dimensional scatter plots of elementary three-dimensional data sets and one high-dimensional data set (see also Figure 6.1 in the next chapter) will be presented to illustrate this claim.
Thereafter, structure preservation will be defined in this chapter to serve as the basis for a new method of visualization. This new concept with regard to the visualization of projected points in a two-dimensional output space is called the generalized U-matrix approach. In the generalized U-matrix approach, similarities between high-dimensional data are represented as valleys, and dissimilarities are represented as mountains or ridges. For the computation of the generalized U-matrix, the generation of the topographic map (see chapter 5.3) and island visualization the CRAN R package GeneralizedUmatrix was used [Thrun/Ultsch, 2017b].

5.1 Examples

In Figure 5.1, the Hepta data set is shown. The Hepta data set [Moutarde/Ultsch, 2005] consists of 7 clusters that are clearly separated by distance, which means that the intracluster distances are small and the intercluster distances are large (for details, see chapter 9). This gives rise to structures that are clearly defined by discontinuity and consequently can be characterized as natural clusters.
Projections of the Hepta data set obtained by applying three of the projection methods introduced in the previous chapter are shown in Figure 5.2: PCA, curvilinear component analysis (CCA) and t-distributed stochastic neighbor embedding (t-SNE). In total, four projections are evaluated, including two t-SNE projections, denoted by *t-SNE (1)* and *t-SNE (2).* PCA yields the best representation of the clusters. With the default parameters, CCA adds excessive gaps around three points. In t-SNE (1), generated using the default parameter settings of t-SNE, the density of the data is overestimated, and wide gaps are added between two points and their corresponding cluster. When one parameter of the t-SNE algorithm is changed, resulting in t-SNE (2), the data clusters are not preserved because many random gaps are added.

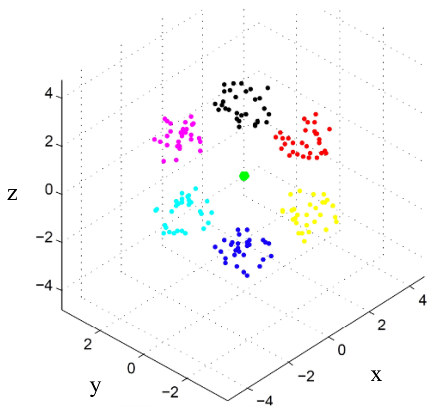

Figure 5.1: The three-dimensional Hepta data set consists of 7 clusters that are clearly separated by distance. One cluster (green) has a higher density. Every cluster is ball-like in shape.

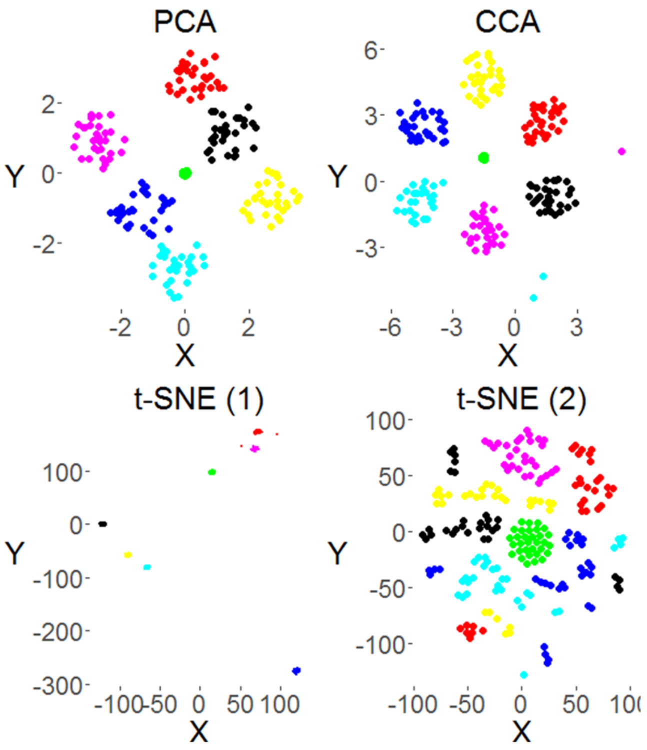

Figure 5.2: Visualizations of four cases of the projection of the Hepta data set into a two-dimensional space generated with [Thrun et al, 2017b].

Top left: PCA projects the data without disrupting any clusters. This is the best-case scenario for a projection method. **Top right**: CCA disrupts two clusters by falsely projecting 3 points. This is the standard-case scenario.

Bottom left: t-SNE does not correctly visualize the density of the data set at all, and one cluster is disrupted through the false projection of two points. Projection methods are often unable to correctly capture the density of data**.** **Bottom right**: When one parameter of the t-SNE algorithm is chosen incorrectly, all clusters are completely disrupted. This is the worst-case scenario for a projection method.

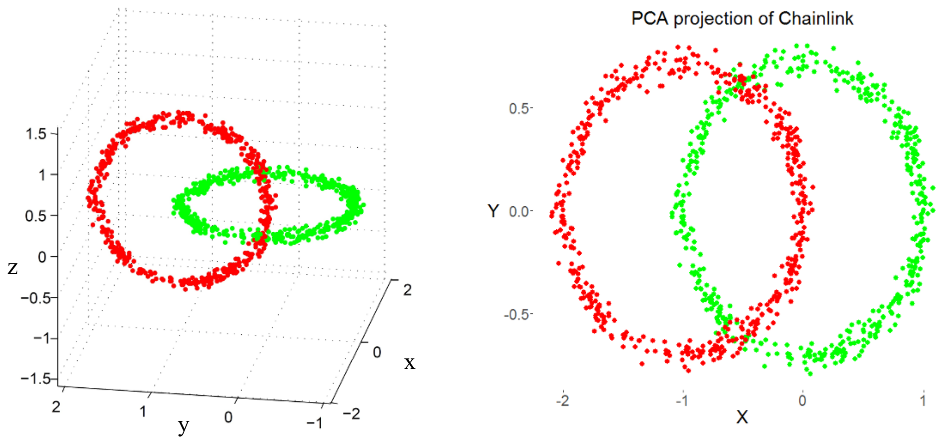

Figure 5.3: Chainlink data set and PCA projection generated with [Thrun et al., 2017]. The projection suffers from local backward projection error (BPE) and forward projection error (FPE) only in two small areas around a low number of points, but the visualization still shows low structure preservation.

The Chainlink data set [Ultsch, 2005c] consists of two clusters in $\mathbb{R}^3$. Together, both clusters form intricate links in a chain and therefore cannot be separated by linear decision boundaries. Both rings are intertwined in $\mathbb{R}^3$ and have the same average distance and density (Figure5.3 left). The data lie on two well-separated manifolds; however, the global proximities contradict the local ones in the sense that the center of each ring is closer to some elements of the other class than it is to elements of its own class (for details, see chapter 9). PCA projection completely fails to preserve the structures in this data set because PCA merely rotates the data set and the discontinuities are not linearly separable.

5.2 Structure Preservation

Let $k > 0$, $k \in \mathbb{N}$, let Γ be a connected graph, and let j be a point in a metric space M; then,

$$H_j(k, \Gamma, M) = \{l \in M \mid G(l, j, \Gamma) \leq k\} \qquad (5.1)$$

is the neighborhood set of j with k as the neighborhood extent, where $G(l, j, \Gamma)$ is the minimum distance among all possible path distances (for details, see chapter 2, Eq. 1).

Suppose that there exists a pair of similar high-dimensional data points $(l_I, j_I) \in I$ such that $(l_I, j_I) \in H(1, \Gamma, I)$. For visualization, the goal of a projection is to match these points to the low-dimensional space R^b; e.g., data points in close proximity should remain in close proximity, and remote data points should stay in remote positions.

Consequently, two kinds of errors exist. The first is forward projection error (FPE), which occurs when similar data points $l \in H_j(1, \Gamma, I)$ are mapped onto far-separated points $l \notin H_j(1, \Gamma, O) \wedge l \in H_j(k > 1, \Gamma, O)$. The second is backward projection error (BPE), which occurs when a pair of closely neighboring positions $l \in H_j(1, \Gamma, O)$ represents a pair of distant data points $l \notin H_j(1, \Gamma, I) \wedge l \in H_j(k > 1, \Gamma, I)$. It should be noted that similar definitions are found in [Ultsch/Herrmann, 2005], for the case of a Euclidean graph; in [Venna et al., 2010], for the case of a KNN graph of binary neighborhoods, where BPE and FPE are referred to as

precision and recall; and in [Aupetit, 2007], for the case of a Delaunay graph, where BPE and FPE are referred to as manifold stretching and manifold compression.
Examples of BPE and FPE are shown in Figure 5.2. The PCA projection of the Hepta data set has a low FPE but a high BPE. The CCA projection has a very low BPE, but three points have high FPEs. The t-SNE (1) projection has a very high FPE, and for the t-SNE (2) projection, both the FPE and BPE are very high.
However, the FPE and BPE are not sufficient measures for evaluating projections if the goal is to estimate the number of clusters or to ensure a sound clustering of the data (e.g., Figure 5.3 right). In such a case, a suitable projection method should be able to preserve discontinuities, which occur in regions of the data space where the probability density function becomes very small. Discontinuities divide a dataset in the input space I into several clusters of similar elements represented by points ([Ultsch/Herrmann, 2005] used a similar definition).
In summary, the quality of structure preservation should be measured based on the preservation of high-dimensional discontinuities as gaps in the two-dimensional output space. Structure preservation refers to the preservation of input-space discontinuities such that no points are allowed to intrude into the corresponding discontinuity regions in the output space.
Let $j \in I$ be an arbitrary point, and let I be projected into O by the function *proj*; then, the projection method *proj* is structure-preserving for a fixed extent $k \in \mathbb{N}$ if

$$proj\colon\ I \to O, H_j(k,\Gamma,I) \mapsto\ H_j(k,\Gamma,O)\ \forall j \in I \qquad (5.2)$$

The direct neighborhoods are preserved if

$$\forall j \in I\colon H_j(1,\Gamma,I) \cap H_j(1,\Gamma,O) = \emptyset \qquad (5.3)$$

The BPE and FPE are acceptable if the quality of structure preservation is high (e.g., Figure 5.3). Notably, the preservation of structure critically depends on the chosen concept of similarity. For example, a multidimensional scaling (MDS) technique may be a suitable projection method if the structure preservation depends only on a Euclidean graph. This is the case for the Hepta data set. By contrast, for the Chainlink data set, a KNN graph with a suitably chosen number of nearest neighbors could yield a better result.
In Chapter 6, it will be demonstrated that many quality criteria exist for evaluating visualizations. Given the definition of structure preservation, it is possible to group these quality measures (QMs) into semantic classes based on graph theory.
In the last section of this chapter, a visualization method with the specific aim of structure preservation is proposed.

5.3 Generating a Topographic Map from the Generalized U*-matrix

In this section I introduce an U*-matrix technique that is generally applicable for all projection methods and can be used to visualize both distance- and density-based structures. This visualization technique is the further development of the idea that the U-matrix can be applied to every projection method [Ultsch/Mörchen, 2006].
In this work, the visualization technique results in a topographic 3D landscape. Here, the requirements are a heavily modified emergent self-organizing map (ESOM) algorithm and a

method of high-dimensional density estimation. Contrary to [Ultsch/Mörchen, 2006], the process of computing the resulting topographic map is completely free of parameter dependence and accessible by simply by downloading the corresponding R package [Thrun/Ultsch, 2017b].

5.3.1 Simplified ESOM

To calculate a U*-matrix for any projection method, a modified ESOM algorithm is required. The first step is the computation of the correct lattice size.
On the x axis, let the lattice begin at 1 and end at a maximal number denoted by Columns C (equal to the number of columns in the lattice); similarly, on the y axis, let the lattice begin at a maximal number denoted by Lines L and end at 1. Then, the first condition is expressed as [Ultsch, 2015]

$$\frac{L-1}{C-1} \approx \frac{|\max(y)-\min(y)|}{|\max(x)-\min(x)|} = \frac{dy}{dy} = \Delta \qquad (I.)$$

The second condition is that the lattice size should be larger than NN[27]:

$$L * C \geq \mathrm{NN} \qquad (II.)$$

The first condition (I.) implies that the lattice size should be as close to equal to the size of the coordinate system as possible. The second condition (II.) is required for emergence in our algorithm. For details, see [Ultsch, 1999]. The resulting equation to be solved is

$$L^2 + L(1+\Delta) - \mathrm{NN} * \Delta \geq 0 \qquad (5.4)$$

which yields

$$L \geq -\frac{1+\Delta}{2} + \sqrt{\left(\frac{1+\Delta}{2}\right)^2 + \mathrm{NN} * \Delta} \qquad (5.5)$$

After the transformation from the projected points[28] $p \in O$ to points on a discrete lattice, the points are called the best-matching units (BMUs) $bmu \in B \subset \mathbb{R}^2$ of the high-dimensional data points j, analogous to the case for general SOM algorithms with $fgrid: O \to B, p \mapsto bmu,$ where *fgrid* is surjective when conditions (i) and (ii) are met.
To develop the algorithm illustrated in Listing 5.1, the idea of [Ultsch/Mörchen, 2006], in which it was suggested to "apply Self-Organizing Map training without changing the best match[ing unit] assignment", was adopted. However, in contrast to [Ultsch/Mörchen, 2006], here, the transformation *fgrid* is defined precisely to calculate the BMU positions and the structure of the lattice is toroidal; i.e., the borders of the lattice are cyclically connected [Ultsch, 1999].
Based on the relevant *symmetry considerations*[29], a simplified version of ESOM (sESOM) is introduced here. No epochs or learning rate are required, because the cooling scheme is defined by a special neighborhood function $h: M \times M \times \mathbb{R}^+ \to [0,1]$.
Let $M = \{m_1, \dots, m_n\}$ be a set of neurons (where m_i are the lattice positions) with the corresponding prototype set $W = \{w_1, \dots, w_n\}$, where dim(W)=dim(I) and #W=#M; then, the neighborhood function h is defined as

[27] In [Ultsch, 1999] the minimum number of 4096 neuros was proposed.
[28] Or DataBot positions on the hexagonal grid of Pswarm (see chapter 8).
[29] See chapter 8 for details.

$$h = \begin{cases} 1 - \frac{d(j,l)^2}{\pi R^2}, & \text{iff } \frac{d(j,l)^2}{\pi R^2} < 1 \\ 0, & \text{else} \end{cases} \qquad (5.6)$$

In sESOM, learning is achieved in each step by modifying the weights in a neighborhood as follows:

$$\Delta w(R) = 1 * h(bmu(j), m_i, R) * (j - w(m_i)) \qquad (5.7)$$

In contrast to [Ultsch/Mörchen, 2006], the algorithm does not require any input parameters, and the resulting visualization is not a two-dimensional gray-scale map but rather a topographic map with hypsometric tints [Thrun et al., 2016a]. The entire algorithm is summarized in Listing 5.1.

```
function (B, I)
    for all $bmu(j) \epsilon B$:
        assign the positions $m_j \in M$ with random weightings $w_j \epsilon W$ on the grid
        assign to each $bmu(j) = m_j$ the weighting $w_j = j \in I$
    end for $bmu(j)$
    for R=Rmax to 1 do
            for all $j \epsilon I$:
                $bmu(j) = \underset{m \in M}{\operatorname{argmin}}\{D(j, w(m))\}$
                    $\Delta w(R, bmu(j)) = h(bmu(j), m_i, R) * (j - w(m_i))$
                    for all $w(m_k) \in h(bmu(l), m_i, R)$
                            $w'(m_k) = w(m_k) + \Delta w(R, bmu(l))$
                    end for $w(m_k)$
            end for $j \epsilon I$
            for all $bmu(j) \epsilon B$:
                    assign to each $bmu(j) = m_j$ the weighting $w_j = j \in I$
    end for R
end function
```

Listing 5.1: sESOM pseudocode algorithm implements a stepwise iteration from the maximum radius Rmax which is given by the lattice size (Rmax = C/6) stepwise with one per step and down to 1. *$w'(m_k)$* indicates that the prototype *$w(m_k)$* of neuron m_k is modified by Eq. 5.7
Additionally, the search for a new best matching unit still is used and these prototypes may change during one iteration. The predefined prototypes are reset to the weights of their corresponding high-dimensional data points after each iteration.

5.3.2 U-Matrix Calculation*

After sESOM projection, the structure of the input data emerges when a visualization technique called U-matrix is applied. A U-matrix represents a folding of the high-dimensional space in which each receptive field is called a U-height. Let *N(j)* be the eight immediate neighbors of $m_j \in M$, and let $w_j \in W$ be the prototype corresponding to m_j; then, the average of all distances between w_j and the other prototypes w_i is called the U-height corresponding to the position m_j:

$$u(j) = \frac{1}{n} \sum_{i \in N(j)} D(w_i, w_j), \qquad n = |N(j)| \qquad (5.8)$$

To explain the visualization technique for the sESOM algorithm, in this section and in section 5.3.3 below, [Thrun et al., 2016a] is cited:

"The U-matrix is the display of values $u(j)$ through proportional intensities of grey shades [Ultsch, 2003a]. By formalizing the displayed structures, [Lötsch/Ultsch, 2014] showed that the U-matrix is an approximation of [the] Voronoi borders of the high-dimensional points in the output space" (see chapter 4.2.0).

Therefore, the generalized U-matrix can be normalized [using] the generalized abstract U-matrix.

"In addition to the U-matrix, [Ultsch, 2003c] introduced the high-dimensional density visualization technique called P-matrix, where P-heights on top of the receptive fields are displayed. The P-height $p(m_i)$ for a position m_i is a measure of the density of data points in the vicinity of $w(m_j)$:

$$p(m_j) = |\{i \in I | D(i, w(m_j)) < r > 0, r \in \mathbb{R}\}| \qquad (5.9).$$

The P-height is the number of data points within a hypersphere of radius r. Here, we choose the interval ϱ of the radius with

$$\varrho \in [median(C(D)), median(A(D))], \qquad (5.10)$$

where D [represents] all input space distances and A(D) is the group A of distances calculated by [the] ABC analysis [Ultsch/Lötsch, 2015]. ABC analysis[30] tries to identify the optimum information that can be validly retrieved by using concepts developed in economic sciences. In particular, [these] concepts are used in the search for a minimum possible effort that gives the maximum yield [Ultsch/Lötsch, 2015]. The distances are divided into three disjoint subsets A, B and C, with subset A comprising [the] largest values ("outer cluster distances"), subset B comprising values where the yield equals the effort required to obtain it, and the subset C comprising [] the smallest values ("inner cluster distances"). We suggest [choosing] the specific radius r [based on] the [ratio] v of [the] inter- versus intracluster distances[,] estimated [as]

$$v = \frac{max(C(D))}{min(A(D))} \qquad (5.11)$$

The radius r is estimated [as] $r = v * p20(D)$, where $p20(D)$ is [the] 20-th percentile of [the] input distances [Ultsch, 2003b]. From this starting point, the user may search interactively for the empirical Pareto percentile [that] defines the radius r (see [the] R package Umatrix).

The combination of a U-matrix and a P-matrix is called [a] U-matrix [Ultsch et al., 2016a]. It can be formalized as [a] pointwise matrix [product]: $U^* = U * F(P)$, where F(P) is a matrix of factors f(p) that are determined through a linear function f on the P-heights p [in] the P-matrix. The function f is calculated so that f(p) = 1 if p is equal to the median and f(p) = 0 if p is equal to the 95-[th] percentile (p95) of the heights in the P-matrix. For p(j) > p95, f(p) = 0, which indicates that j is well within a cluster and results in [a height of zero] in the U*-matrix." [Thrun et al., 2016a]*

5.3.3 Topographic Map with Hypsometric Tints

The U*-matrix visualization technique produces a topographic map with hypsometric tints [Thrun et al., 2016a]. Hypsometric tints are surface colors that represent ranges of elevation [Patterson/Kelso, 2004]. Here, a specific color scale is combined with contour lines.
The color scale is chosen to display various valleys, ridges and basins: blue colors indicate small distances (sea level), green and brown colors indicate middle distances (low hills), and white colors indicate large distances (high mountains covered with snow and ice). Valleys and

[30] For usage see CRAN R package ABCanalysis [Thrun et al. 2015].

basins represent clusters, and the watersheds of hills and mountains represent the borders between clusters (Figure 5.1 and Figure 5.4).
The landscape consists of receptive fields, which correspond to certain U*-height intervals with edges delineated by contours. This work proposes the following approach (see [Thrun et al., 2016a, p. 10]): First, the range of U*-heights is split up into intervals, which are assigned uniformly and continuously to the color scale described above through robust normalization [Milligan/Cooper, 1988]. In the next step, the color scale is interpolated based on the corresponding CIELab color space [Colorimetry, 2004]. The largest possible contiguous areas corresponding to receptive fields in the same U*-height intervals are outlined in black to form contours. Consequently, a receptive field corresponds to one color displayed in one particular location in the U*-matrix visualization within a height-dependent contour. Let u(j) denote the U*-heights, and let q01 and q99 denote the first and 99-th percentiles, respectively, of the U*-heights; then, the robust normalization of the U*-heights u(j) is defined by

$$u(j) = \frac{u(j) - q01}{q99 - q01} \qquad (5.12)$$

The number of intervals in is defined by

$$\frac{1}{in} = \frac{q01}{q99} \qquad (5.13)$$

The resulting visualization consists of a hierarchy of areas of different height levels represented by corresponding colors (see Figure 5.4). To the human eye, the visualization using the generalized U-matrix tool is analogous to a topographic map; therefore, one can visually interpret the presented data structures in an intuitive manner. In contrast to other SOM visualizations, e.g., [K. Tasdemir/Merenyi, 2009], this topographic map presentation enables the layman to interpret sESOM results.
The use of a toroidal map for sESOM computations necessitates a tiled landscape display in the interactive U-matrix tool [Thrun et al., 2015], which means that every receptive field is shown four times. Consequently, in the first step, the visualization consists of four adjoining images of the same U-matrix [Ultsch, 2003a] (the same is true for the U*-matrix). To obtain the 3D landscape (island[31]), [Thrun et al., 2016a, p. 10] proposed to rectangularly cut the tiled U*-matrix visualization as follows.
Let v_{Lines} and $v_{Columns}$ be the vectors of the row and column sums, respectively, of the U*-heights, and let b_{Lines} ($b_{Columns}$) be the number of BMUs in the corresponding row line of v_{Lines} ($v_{Columns}$); then, we define the upper border as up = max($v_{Lines}/f(b_{Lines})$), the left border as lb = max($b_{Columns}/f(v_{Columns})$) and the other two borders based on the length and width of the U*-matrix, where the vector *f(b)* is the sum $f(b) = \hat{b} + b + \breve{b}$ with
$\hat{b} = (b_n, b_1, \dots, b_{n-1})$ and $\breve{b} = (b_2, \dots, b_{n+1})$ for a toroidal lattice. For better comprehensibility, see the axes in [Thrun et al., 2016a, p. 14, Fig. 1], which are defined from one to $max(Lines)$ and from one to $max(Columns)$.

[31] An island can be also cut interactively (or the the cutting may be improved) and thus may not be rectangular

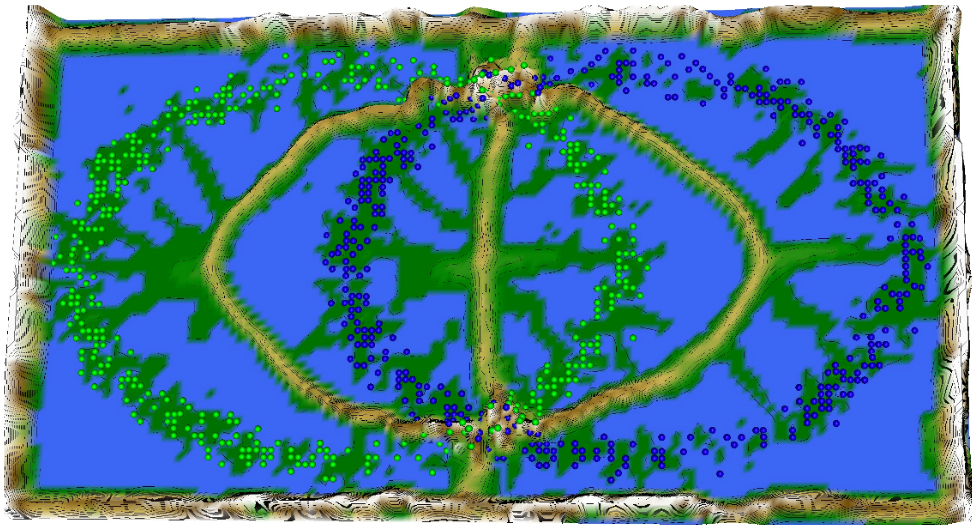

Figure 5.4: Topographic map of the PCA projection of the Chainlink data set. The discontinuities between the clusters are misrepresented.

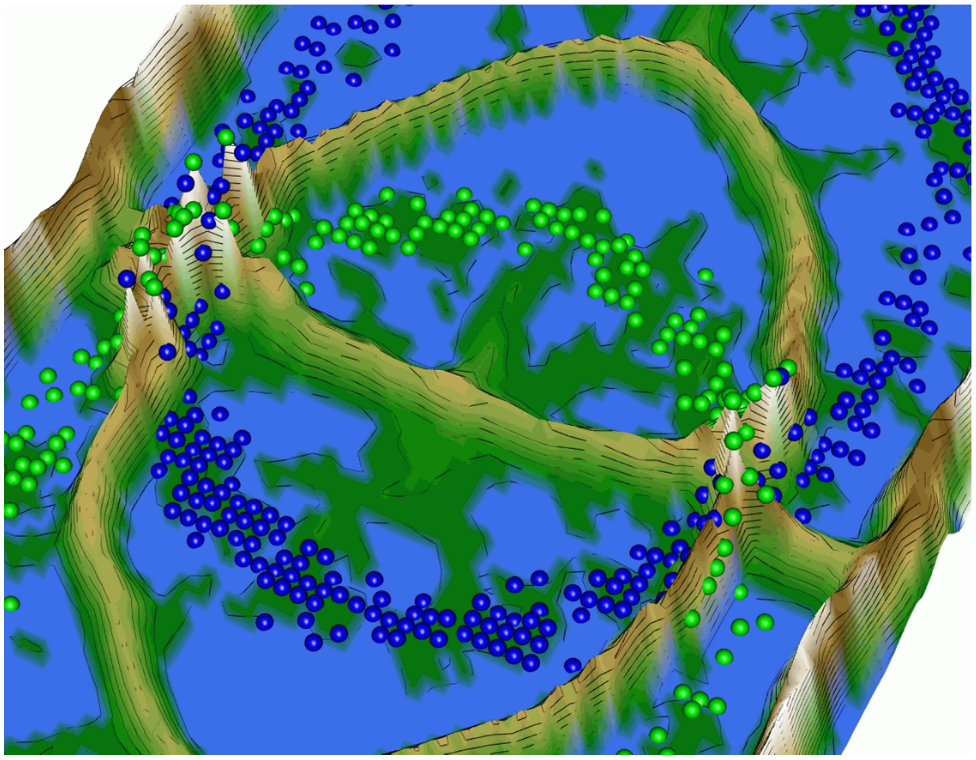

Figure 5.5: Zoomed-in view of the misrepresentation of the discontinuities in the PCA projection of the Chainlink data set to better visualize the BPE and FPE.

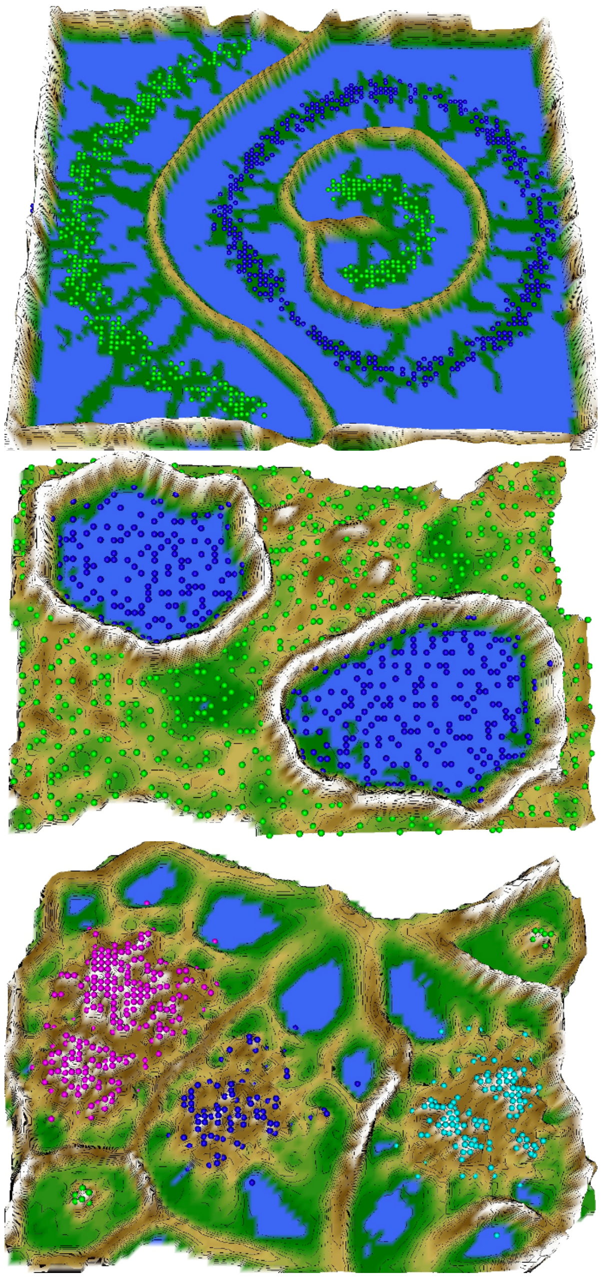

Figure 5.6: Topographic maps can depict the discontinuities in high-dimensional data sets: clusters lie in valleys and are separated by hills. However, the introduction of spurious gaps between projected points (the disruption of clusters) cannot be seen using this approach.
Top: topographic map of CCA projection [Demartines/Hérault, 1995] of the Chainlink data set. **Middle**: topographic map of ESOM projection [Ultsch, 1999] of the Atom data set.
Bottom: island of NeRV projection [Venna et al., 2010] of the leukemia data set. All results are trial-dependent because the projection methods are stochastic. Sometimes, the annealing scheme (in CCA or ESOM) or the random initialization process (in NeRV) fails.

5.3.4 Limitations

The generalized U*-matrix visualization by a topographic map is capable of visualizing BPEs and FPEs. For example, this is shown in Figure 5.5. The projected points in the output space with low BPE/FPE values lie in sea regions. If the BPE/FPE around a projected point is high, then the visualization generates a mountain at this point (Figure 5.5). However, the topographic map has certain limitations (Figure 5.6). When the default parameters in CCA are used to analyze the Chainlink data set (see [Thrun et al., 2017]) or when the default ESOM parameters ([Thrun et al., 2016b]) are used to analyze the Atom data set, clusters are sometimes disrupted because additional gaps are added that cause points to intrude into the discontinuity regions between clusters.

Another question that arises in this chapter from the examples of the CCA and ESOM projections of the Chainlink and Atom data sets, respectively, in Figure 5.6 is the question of how to handle stochastic projection methods in which the visualization is trial-dependent. The annealing schemes used in the ESOM and CCA algorithms may be relevant here. The annealing process depends on certain parameters and may not yield structure-preserving projections, as shown in the examples in Figure 5.6 The Neighborhood Retrieval Visualizer (NeRV) projection of the leukemia data set presented in Figure 5.6 further illustrates the problem of the correct choice of parameters, which is typically very challenging. In this case, the NeRV projection is sensitive to the initialization parameters, especially to the seed used for the random number generator. In chapter 9, an additional example will be presented to demonstrate that NeRV requires the weighting between precision and recall to be correctly chosen for high-dimensional structures to be preserved.

Hence, the next chapter will focus on the search for a QM that may be able to measure structure preservation instead of attempting to visualize it.