# DimenFix

## A novel meta-strategy to preserve user-defined data values on dimensionality reduction layouts

Han, Zixuan; van der Hoorn, Diede; Höllt, Thomas; Luo, Qiaodan; Christino, Leonardo; Milios, Evangelos; Paulovich, Fernando V.

**DOI**
10.1016/j.cag.2025.104231

**Publication date**
2025

**Document Version**
Final published version

**Published in**
Computers and Graphics

**Citation (APA)**
Han, Z., van der Hoorn, D., Höllt, T., Luo, Q., Christino, L., Milios, E., & Paulovich, F. V. (2025). DimenFix: A novel meta-strategy to preserve user-defined data values on dimensionality reduction layouts. *Computers and Graphics*, *130*, Article 104231. https://doi.org/10.1016/j.cag.2025.104231

Computers & Graphics 130 (2025) 104231

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on EuroVA 2024 SI

# DimenFix: A novel meta-strategy to preserve user-defined data values on dimensionality reduction layouts

Zixuan Han [a], Diede van der Hoorn [b], Thomas Höllt [a], Qiaodan Luo [c], Leonardo Christino [b], Evangelos Milios [c], Fernando V. Paulovich [b,*]

[a] Faculty of Electrical Engineering Mathematics and Computer Science, TU Delft, Van Mourik Broekmanweg 6, 2628 XE Delft, The Netherlands
[b] Department of Mathematics and Computer Science, TU Eindhoven, De Zaale, Metaforum, PO Box 513, 5600 MB Eindhoven, The Netherlands
[c] Faculty of Computer Science, Dalhousie University, 6050 University Avenue, PO BOX 15000, B3H 4R2 Halifax, NS, Canada

ARTICLE INFO

Dataset link: https://github.com/fpaulovich/dimensionality-reduction/

*Keywords:*
Dimensionality reduction
Visualization

ABSTRACT

Dimensionality Reduction (DR) methods have become essential tools for the data analysis toolbox. Typically, DR methods combine features of a multivariate dataset to produce dimensions in a reduced space, preserving some data properties, usually pairwise distances or local neighborhoods. Preserving such properties makes DR methods attractive, but it is also one of their weaknesses. When calculating the embedded dimensions, usually through non-linear strategies, the original feature values are lost and not explicitly represented in the spatialization of the produced layouts, making it challenging to interpret the results and understand the features' contributions to the attained representations. Some strategies have been proposed to tackle this issue, such as coloring the DR layouts or generating explanations. Still, they are post-processes, so specific features (values) are not guaranteed to be preserved or represented. This paper proposes *DimenFix*, a novel meta-DR strategy that explicitly preserves the values of a particular user-defined feature or external data (not used to generate a layout) in one of the embedded axes. *DimenFix* can be used to preserve ordinal (e.g., numerical measures) and nominal (e.g., labels) values and works with virtually any gradient-descent DR method. It requires minimum changes to the underlying DR technique, running in linear time considering the number of data instances. In our results, involving Force Scheme and t-SNE adaptations, *DimenFix* was capable of representing features without heavily impacting distance or neighborhood preservation, allowing for creating hybrid layouts that join characteristics of scatter plots and DR methods.

## 1. Introduction

Demand for visualizing and interpreting high-dimensional datasets has rapidly increased in recent years. One of the most popular strategies to interpret such datasets is projecting them to a lower-dimensional space (usually 2D or 3D) while reproducing the similarity relationships among high-dimensional data instances in the produced layout. This process is usually called Dimensionality Reduction (DR) and has been extensively used by the visualization community in most visual analytics solutions, and by many research fields to support multiple different analytical applications, including protein folding [1,2], sensors and biosensors [3–5], single-cell transcriptomics [6], or drug design [7], just to mention a few.

Common to most DR techniques is that the embedded dimensions are defined as combinations of the input data features. Consequently, understanding how feature values contribute to the produced layouts can be challenging since the positions of the projected data instances are typically influenced by all input features, in some cases, through non-linear combinations. Some strategies have been devised to allow for such interpretation, as it is one of the most important DR-based data analysis tasks [8]. Features can be mapped to axes (lines) in the produced layouts to represent the influence of each feature [9]. Color can represent the values of a feature or external data (e.g., a class) [10]. More advanced approaches can also be employed, for instance, contrastive [11] or feature importance [12–14] analyses to identify the contributions of features to the formation of groups of instances. Although effective methods to interpret a DR layout, they are post-process strategies, so they cannot guarantee that specific features (values) are preserved and ordered in the final layout. Therefore, the spatial position, used in standard scatterplots to represent the original features of the data, is not available as the $x$ and $y$ components of the

* Corresponding author.
*E-mail addresses:* Z.Han-2@student.tudelft.nl (Z. Han), d.p.m.v.d.hoorn@tue.nl (D. van der Hoorn), T.Hollt-1@tudelft.nl (T. Höllt), qd398166@dal.ca (Q. Luo), l.christino@tue.nl (L. Christino), eem@cs.dal.ca (E. Milios), f.paulovich@tue.nl (F.V. Paulovich).

https://doi.org/10.1016/j.cag.2025.104231
Received 15 January 2025; Received in revised form 22 April 2025; Accepted 23 April 2025
Available online 22 May 2025

embedding (for 2D layouts) do not have a clear connection with the input features.

This paper proposes *DimenFix*, a novel meta-dimensionality reduction strategy that addresses this limitation by explicitly encoding the values of a particular feature (e.g., dataset attribute, class, time, or ranking) as position in one of the embedded axes. To achieve this, *DimenFix* maps such a feature into one of the embedded axes and controls the degree of freedom with which the embedded points' positions can move on that axis. We present different approaches to support this process depending on the type of values to be preserved, whether it is ordinal or nominal. We also show results of adapting two different DR techniques, Force Scheme [15] and t-SNE [16], to employ the *DimenFix* strategy, although it can be used in combination with virtually any gradient-descent method, such as UMAP [17]. *DimenFix* is minimally intrusive, requires only small modifications to the underlying DR method, and its computational complexity is linear to the number of data instances, thus having a low impact on the overall running time. Our experiments show that *DimenFix* can represent features in the final layout without heavily impacting distance or neighborhood preservation quality. The code for *DimenFix*, Force Scheme, and t-SNE adaptations can be found at https://github.com/fpaulovich/dimensionality-reduction/.

## 2. Related work

Dimensionality Reduction (DR) techniques in the visualization field are defined as techniques that map instances from a high-dimensional space, the input dataset, to points in a lower-dimensional space, the visual representation, usually 2D or 3D, seeking to preserve high-dimensional similarity relationships [18,19]. Different taxonomies are used to classify such techniques depending on the different aspects of the data to be preserved or the strategy used for the mapping. A common taxonomy classifies the techniques into local and global [16]. While global methods, such as Multidimensional Scaling (MDS) [20], Force Scheme [15], or Part-Linear Multidimensional Projection (PLMP) [21], look to preserve the overall pairwise distance between data instances, local techniques, such as t-Distributed Stochastic Neighbor Embedding (t-SNE) [16], Uniform Manifold Approximation and Projection (UMAP) [17], and Least Squares Projection (LSP) [22], seek to preserve local neighborhoods.

Common to these techniques is the fact that they combine all the input high-dimensional data features or dimensions to compose the final visual representation, using linear or non-linear strategies. For linear strategies, the interpretation of the connection between an axis in a layout and the original data features can, to an extent, be performed since axes are linear combinations of the input features. However, for non-linear strategies, this connection is usually not meaningful since points are individually mapped instead of axes. To address this issue, different strategies have been suggested to allow for interpretation considering the contribution of input features. Mapping the original dataset features/dimensions to segments in the DR layout has been suggested [9], in which shorter segments represent small feature contributions, and the direction of the segments indicates how the related feature varies in the layout. Color is also a popular choice, where data instances (or points in the visual representation) are colored using the values of a feature or class to map additional information [10]. Color has also been used to represent distance distortions calculated to a reference point by coloring the visual representation background [23], or considering the overall distance preservation either by coloring points [24] or through a more sophisticated coloring scheme [25]. Although the resulting layouts can be used effectively to identify artifacts of the DR approximation, they cannot be used to convey information about dataset features, which is a common limitation of the visual quality metric strategies such as [26,27], since this is not their focus.

More advanced strategies have also been suggested to describe the importance of features in DR layouts using Shapley values [12], by contrastive analysis [11], or by detecting the most important features to describe groups [13,14,28]. Although effective methods to interpret a DR layout, they are post-processes, and thus defining the contribution of a specific input feature or requiring it to contribute to the layout's spatialization remains challenging since what is visible is constrained by the DR method, and features can have near-zero contribution [9]. Neither are there guarantees that ordered values of a feature will follow an order in the final layout, making the interpretation of ordered values, such as numerical data features or external information, such as time or ranking, very hard or even impossible.

Our approach, *DimenFix*, focuses on solving this issue, allowing for the order of a specific data feature or any external information to be explicitly represented while still retaining the original distances and local neighborhoods of the high-dimensional space as much as possible in the produced layout.

## 3. Methodology

### 3.1. Overview

*DimenFix* is a strategy that can be applied to modify virtually any Gradient-Descent (GD) Dimensionality Reduction (DR) method, such as t-SNE [16], Force Scheme [15] or UMAP [17], to allow for the preservation of a given, ordered, e.g., data feature, or unordered, e.g., instances' labels, one-dimensional attribute by mapping it to one of the visual layout axes, from now on called *fixed axis*. Typically, any GD-DR method is free to update the values of all the $n$ embedded axes in the optimization iterations. *DimenFix* changes this by allowing only $(n-1)$ axes to change freely, while constraining the movement of the fixed axis.

In more formal terms, let $X = \{x_1, \ldots, x_N\} \in \mathbb{R}^m$ denote the input dataset, where $x_i = (x_i^1, \ldots, x_i^m), x_i^j \in \mathbb{R}, 1 \leq j \leq m$ are the coordinates of a $m$-dimensional data instance and $x^j = (x_1^j, \ldots, x_N^j)$ is the $j$th feature or dimension of $X$. Also, let $Y = \{y_1, \ldots, y_N\} \in \mathbb{R}^n$ be the final embedding, where $y_i = (y_i^1, \ldots, x_i^n), y_i^j \in \mathbb{R}, 1 \leq j \leq n$ is the mapping of $x_i$ to the $n$-dimensional (visual) embedding space, and let $V = \{v_1, v_2, \ldots, v_N\}$ be the values to be mapped to the fixed axis, that is, the selected feature of $X$ or the external metadata, such as labels for the data instances, to be preserved.

To adapt a GD-DR technique to use *DimenFix*, the initial embedding configuration is set so that the fixed axis receives the values in $V$, and the others are randomly initialized — the free axes could also be initialized deterministically, for instance, using PCA. Without loss of generality, in this initialization, we set $y_i^1 = v_i, 1 \leq i \leq N$ and $y_i^j \sim U([0,1]), 1 \leq i \leq N, 2 \leq j \leq n$ where $U([0,1])$ denotes a random number uniformly sampled in $[0,1]$. After that, at every $k$ iterations of the optimization, the gradient $\nabla E(Y)$ is calculated, where $E(Y)$ is the embedding cost function, the new embedded coordinates $\overline{Y}$ are computed, and the resulting embedding $Y$ at such an iteration is created by bounding the fixed axis to a range in the visual layout while the other $(n-1)$ axes are freely updated. Algorithm 1 describes this process. Notice that, considering the overall process, *DimenFix* only needs access to the updated projection coordinates resulting from a gradient descent iteration (line 6 of the algorithm) to execute a pull. In that sense, it is agnostic to the gradient descent strategy. Algorithm 1 defines a simple gradient descent process for illustration, but other improvements or modifications could be included, such as a more sophisticated termination criterion, without implying any changes to *DimenFix*. We only need to guarantee that a final pull is executed at the end of the process (line 19 of the algorithm).

In this process, two main components need to be defined, namely, the *range of movement* which each data instance can occupy on the fixed axis and the *pulling mode* strategy to bound the movement of the instances inside a range. These two components, composing the core of *DimenFix*, are discussed next.

**Algorithm 1:** General *DimenFix* pseudocode for gradient-descent like methods.

**Data:** $X = \{x_1, \dots, x_N\}$: dataset
$\eta$: learning rate
$max$: maximum number of iterations
$V = \{v_1, \dots, v_N\}$: feature/metadata to be preserved
$feature = \{ordinal, nominal\}$: feature/metadata type
$k$: number of iterations to pull the points
$\alpha$: Overlap control between ranges
**Result:** $Y$: final embedding

$y_i^1 \leftarrow v_i, 1 \le i \le N$
$y_i^j \sim U([0,1]), 1 \le i \le N, 2 \le j \le n$
$\hat{Y} \leftarrow compute\_preferred\_positions(Y, V)$
$R \leftarrow compute\_ranges(\hat{Y}, \alpha)$
**while** $it < max$ **do**
  $\overline{Y} \leftarrow Y - \eta \nabla E(Y)$
  **if** $(it \mod k) = 0$ **then**
    **if** $feature = nominal$ **then**
      $\overline{Y} \leftarrow rotate(\overline{Y})$
      $\hat{Y} \leftarrow compute\_preferred\_positions(\overline{Y}, V)$
      $R \leftarrow compute\_ranges(\hat{Y}, \alpha)$
    **end**
    $\overline{Y} \leftarrow scale(\overline{Y})$
    $\overline{Y} \leftarrow pull(\overline{Y}, \hat{Y}, R)$
  **end**
  $Y \leftarrow \overline{Y}$
  $it = it + 1$
**end**
$Y \leftarrow pull(\overline{Y}, \hat{Y}, R)$

### 3.2. Range of movement

To each point $y_i$ a position $\hat{y}_i \in \hat{Y}$ in the fixed axis and a range $R_i = [\hat{y}_i - \alpha \times B_i, \hat{y}_i + \alpha \times T_i]$ are associated, defining the preferred position for $y_i$ and the section of the fixed axis it can occupy, where $B_i$ defines the lower limit and $T_i$ the upper limit of the range around $\hat{y}_i$. In our approach $\alpha \ge 0$; when $\alpha = 0$, $y_i$ is fixed to the position $\hat{y}_i$; when $0 < \alpha \le 1$, $y_i$ can move in $[\hat{y}_i - B_i, \hat{y}_i + T_i]$; otherwise, $y_i$ can get outside of such an interval. Defining how $\hat{y}_i$, $B_i$, and $T_i$ are set depends on whether the values $V = \{v_1, \dots, v_N\}$ used in the fixed axis are ordinal (where $>$ and $<$ operations apply), e.g., a ranking, or nominal (where only $=$ and $\neq$ operations apply), e.g., labels.

#### 3.2.1. Ordinal values

Without losing generality, let $v_1 \le v_2 \le \cdots \le v_N$, then $\hat{y}_i = v_i$, and $B_i = (v_i - v_{i-1})/2$ and $T_i = (v_{i+1} - v_i)/2$. In this way, the preferred positions $\hat{Y}$ are the values to be preserved given by $V$ and the intervals between consecutive values do not overlap while all the available space on the fixed axis is used, and an interval $R_i$ contains the value $v_i$ that should be preserved in the fixed axis. The only exceptions to this range definition are the first and last values of $V$. For $v_1$ and $v_N$, $T_1$ and $B_N$ do not change, but since there are neighbors only on one side of the range, it is symmetrized, so that $B_1 = T_1$ and $T_N = B_N$. In summary, the size and position of the ranges assigned to each point $y_i$ are defined by the order and magnitude of the values in $V$.

#### 3.2.2. Nominal values

For nominal values, the strategy for defining ranges and positions is more involved since such values cannot be naturally ordered. Consider that $V$ contains $K$ unique values, for instance, $K$ different labels. In our strategy, suppose $Y = Y_1 \cup \cdots \cup Y_K$ is composed of $K$ disjoint partitions defined by the $K$ unique values in $V$ and that $\Phi(Y_i)$ returns the order in $[1, K] \in \mathbb{N}$ of a partition $Y_i$, and that, without loss of generality, $\Phi(Y_1) \le \Phi(Y_2) \le \cdots \le \Phi(Y_K)$, the position of $y_i \in Y_j$ is defined as $\hat{y}_i = {}^{(2\times\Phi(Y_j)-1)}/_{(2\times K)}$, and the minimum $B_i$ and maximum $T_i$ values of the range are defined as $B_i = {}^{(\hat{y}_i-1)}/_{(2\times K)}$ and $T_i = {}^{(\hat{y}_i+1)}/_{(2\times K)}$. So that all the instances belonging to the same partition receive the same preferred position and range.

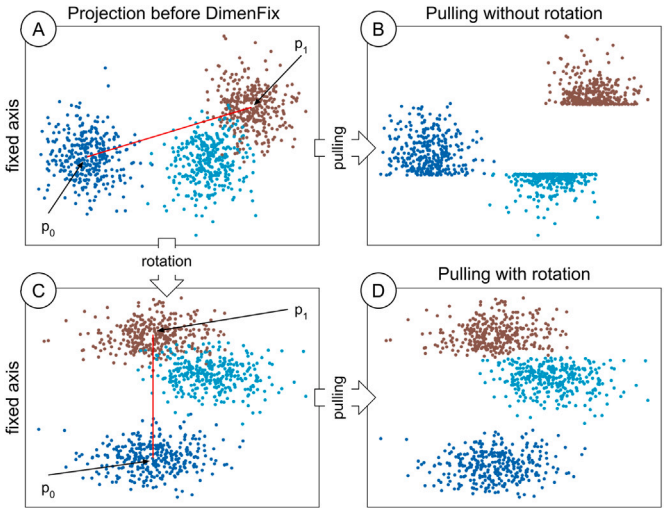


**Fig. 1.** Rotation process to solve partitions' misalignment problems. Points are overlapped close to the range boundaries in case a projection with poorly aligned partitions with the fixed axis (A) is pulled (B). Better results are obtained if a rotation is executed to first align the partitions' separation with the fixed axis (C) before the pulling is executed (D).

In this process, the missing element is the strategy to order the partitions $Y_i$. In the first execution of *DimenFix*, $\Phi$ defines a random order for the partitions; however, as the GD process progresses, $\Phi$ defines an ordering for the partitions so that similar partitions are close in the ordering. Taking advantage of the fact that DR techniques focus on preserving similarity structures of the original space, in this process, we first calculate the mean $m_i$ of each partition $Y_i$ considering the fixed axis in the current layout $Y$, that is, $m_i = \sum_{y_j \in Y_i} (y_j^1 / |Y_j|)$. After that, the partitions are ordered based on their means, and $\Phi$ returns the position of each partition in this ordering. This strategy ensures that the ordering of the nominal values on the produced layout $Y$ considers, up to an extent, the similarity among the groups of instances with the same nominal value.

Extreme cases may happen when the partitions $Y_1, \dots, Y_K$ are misaligned with the fixed axis; for instance, when the partitions overlap with respect to the fixed axis. When this happens, layouts are usually of poor quality, even if the partitions are well separated in the visual layout by considering the free axes. To prevent this from happening, before ordering the partitions, we rotate the layout $Y$ to align the partitions' separation with the fixed axis. Fig. 1 illustrates this process where three reasonably well-separated partitions overlap on the fixed $y$-axis. Based on the partitions' centroids, we first find the two farthest partitions in $Y$ (dark blue and brown). Considering these two partitions, suppose that $p_0$ is the centroid of the partition with the minimum coordinate in the fixed axis, and $p_1$ is the centroid of the other partition (Fig. 1(A)). First, $Y$ is translated so that $p_0$ is at the origin and then $Y$ is rotated so that the vector $\vec{v} = (p_1 - p_0)$ (red segment) is aligned (parallel) with the fixed axis (Fig. 1(C)). The misalignment issue in case the rotation is not executed is represented in Fig. 1(B) after a Gaussian pulling is executed (see Section 3.3.2). Points are highly overlapped close to the boundary between ranges. With the rotation (Fig. 1(D)), this problem is reduced, and the overall point distribution is better preserved.

### 3.3. Pulling mode

As the GD process progresses, some points $y_i$ may get out of the designated range $R_i = [\hat{y}_i - \alpha \times B_i, \hat{y}_i + \alpha \times T_i]$. So, at every certain number of iterations of the GD process, such points are pulled back to their ranges. We developed 3 different modes for pulling points depending on the complexity and final application. They are explained in the following subsections.

#### 3.3.1. Clipping pulling mode

The simplest pulling mode is *Clipping*. The general idea is to pull a point $y_i$ to the limits of $R_i$ if it gets outside of the range. Let $\overline{y}^1$ be the current calculated values for the fixed axis. In the clipping mode, the values $y_i^1, 1 \leq i \leq N$ are updated to

$$y_i^1 = \begin{cases} \hat{y}_i - \alpha \times B_i & \overline{y}_i^1 < \hat{y}_i - \alpha \times B_i \\ \hat{y}_i + \alpha \times T_i & \overline{y}_i^1 > \hat{y}_i + \alpha \times T_i \\ \overline{y}_i^1 & \text{otherwise,} \end{cases} \tag{1}$$

composing the function *pull*() of line 14 of Algorithm 1.

Although a very simple process, the clipping mode is preferred when the values $V$ to be fixed are ordinal and the intention is to produce a layout that preserves such values as much as possible. This can be achieved by reducing the value of $\alpha$, since with $\alpha = 0$, the fixed axis will preserve the same values in $V$.

#### 3.3.2. Gaussian pulling mode

The clipping pull mode allows moving a point on the fixed axis within a certain range. However, the hard threshold may cause the movement of the points in the fixed axis to be inconsistent when compared to the other axes. The *Gaussian* pull mode produces better results by allowing points to go beyond the range threshold using a Gaussian function. Under this mode, we use a Gaussian to weight the moving force. The farther the point is from its preferred position $\hat{y}_i$, the harder it is to move. As a result, some points may slightly move out of the predefined range $R_i$, resulting in a more consistent movement.

Let $\Delta_i$ be the difference between the preferred and current positions of the fixed axis for the point $y_i$, that is, $\Delta_i = \hat{y}_i - \overline{y}_i^1$ – without loss of generality, we always fix the first axis of the produced layout. Since when $\Delta_i = 0$ the moving force should be maximum (equal to 1), the mean in the Gaussian should be $\mu_i = 0$, and the Gaussian weighting function can be defined as

$$f(\Delta_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{\Delta_i^2}{2\sigma_i^2}\right), \tag{2}$$

so that $f(\Delta_i)$ defines a fraction of the actual moving distance along the fixed axis for the point $y_i$.

To calculate $\sigma_i$, consider a user-defined confidence interval $0 < CI < 1$. First, we use the $z$-score table to find the $z$-score ($z$) where $^{(1-CI)}/_2$ is the area on the left side of the significance interval. This $z$-score refers to how far a value is from the mean in a standard normal distribution, and we use it as a reference to change the shape of the Gaussian function. With $z$, a maximum interval spread of

$$\beta_i = \begin{cases} |\hat{y}_i - B_i| & \overline{y}_i^1 < \hat{y}_i \\ |\hat{y}_i - T_i| & \text{otherwise,} \end{cases} \tag{3}$$

is defined where $\overline{y}_i^1$ is the current calculated value for the fixed axis, and, based on the condition $\mu_i = 0$, we use the $z$-score equation ($z = {}^{(\beta_i - \mu_i)}/_{\sigma_i}$) to calculate $\sigma_i$ as

$$\sigma_i = \frac{\beta_i}{z}. \tag{4}$$

Knowing that when $\Delta_i = 0$, $f(\Delta_i)$ reaches its maximum, we can adapt $f(\Delta_i)$ to weight how much the distance ($\Delta_i$) calculated by the GD-DR method should be used to move a point by bounding $f(\Delta_i)$ to $[0, 1]$. For this, we divide $f(\Delta_i)/f(0)$, resulting in

$$\hat{f}(\Delta_i) = \frac{\frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{\Delta_i^2}{2\sigma_i^2}\right)}{\frac{1}{\sigma_i\sqrt{2\pi}} \exp(0)} = \exp\left(-\frac{\Delta_i^2}{2\sigma_i^2}\right), \tag{5}$$

and apply such weighting scheme to reduce the distance $\Delta_i$ a point $y_i$ has moved outside its range using

$$y_i^1 = \hat{y}_i + \left(\Delta_i \times \exp\left(-\frac{\Delta_i^2}{2\sigma_i^2}\right)\right). \tag{6}$$

This pulls the point $y_i$ closer to its preferred position $\hat{y}_i$ with more force as it gets farther from it, and defines the *pull*() function on line 14 of Algorithm 1.

#### 3.3.3. Rescaling pulling mode

The *Gaussian* pulling mode improves over the *Clipping* mode by reducing the number of points that accumulate close to the limits of the ranges $R_i$ assigned to the points, especially when $V$ contains nominal values. However, the shape of the groups of points sharing the same nominal values is affected. This gives rise to the last type of pulling mode, the *Rescaling*. The rescaling mode is specifically for cases where $V$ is nominal. The idea is to normalize each disjoint partition $Y_1 \cup \cdots \cup Y_K = Y$ that contains points sharing the same nominal value in $V$ to fit within the designed range.

Considering that $min_j$ and $max_j$ are the minimum and maximum values on the fixed axis for the points in the partition $Y_j$, and that $R_i = [\hat{y}_i - \alpha \times B_i, \hat{y}_i + \alpha \times T_i]$ is the range assigned to the points in $y_i \in Y_j$, the values $y_i^1, y_i \in Y_j$ are updated to

$$y_i^1 = (\hat{y}_i - \alpha \times B_i) + \frac{\overline{y}_i^1 - min_j}{max_j - min_j}(\alpha \times (B_i + T_i)) \tag{7}$$

where $\overline{y}_i$ is the current computed position for the point $y_i$, constituting the function *pull*() of line 14 of Algorithm 1.

### 3.4. Scaling

Before pulling, the embedding coordinates are translated and rescaled (function *scale()* on line 13 of Algorithm 1) to match the defined intervals ($R_i$). This is necessary to avoid distortions, especially when the values $V$ to be fixed are nominal, so discrepancies in magnitude between the fixed and the free axes are minimized. For ordinal values, rescaling may not be necessary, but since *DimenFix* does not have control over the gradient-descent process, depending on the internal implementation of the DR technique, distortions may happen, not only due to scaling but also due to translations. In this process, we first center the current projection $\overline{Y} = \overline{Y} - {}^1/_N \sum_{\overline{y}_i \in \overline{Y}} (\overline{y}_i)$, and get the maximum ($max_{range}$) and minimum ($min_{range}$) values in the ranges $R_i$ and the maximum ($max_{axis}$) and minimum ($min_{axis}$) values on the fixed axis. Then $\overline{Y}$ is scaled by $\sigma = {}^{(max_{range} - min_{range})}/_{(max_{axis} - min_{axis})}$ and translated to the range, resulting in $\overline{Y} = (\overline{Y} \times \sigma) + (max_{range} + min_{range})/2$. This guarantees that the projection coordinates have similar magnitudes and are "close" to the values ($V$) *Dimenfix* is aimed to preserve before the pulling.

Since scaling is necessary, if $V$ is composed of non-quantitative ordinal values, before the process starts, they need to be transformed into quantitative values to compose $\hat{Y}$, the preferred positions. A straightforward strategy would be to map the order of the elements in $V$ into sequential integers. In other words, and without loss of generality, suppose $v_1 < v_2 < \cdots < v_N, \forall\ v_i \in V$, the transformation could be $\Psi(v_i) = i$. When $V$ is nominal, order and magnitude are irrelevant since these are adjusted by *DimenFix* (see Section 3.2) during the GD process.

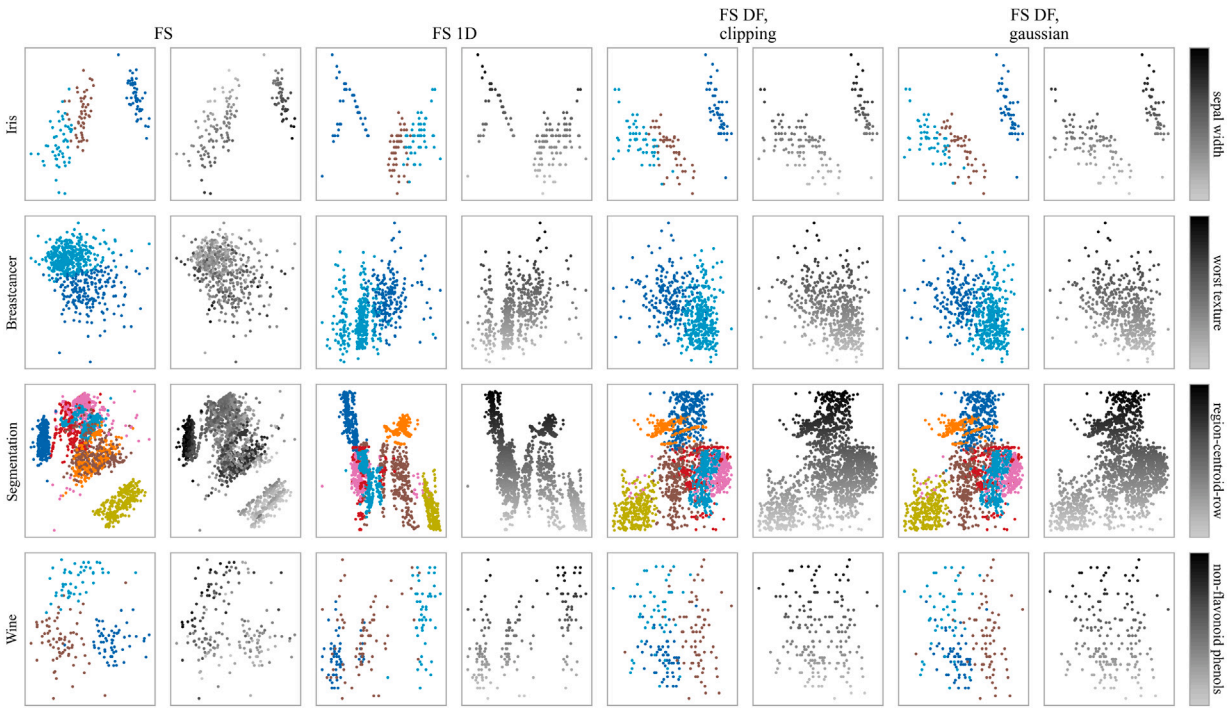


**Fig. 2.** Force Scheme and Force Scheme *DimenFix* adaptation layouts fixing datasets features (ordinal) to the *y*-axis. When particular features are mapped to color, it is possible to see that they do not smoothly vary on the Force Scheme layouts (from small to large values). Hence, reasoning about feature values and point positions is challenging. On *DimenFix* projections, this can be attained by fixing such features in one of the projection axes.

## 4. Results

In this section, we discuss quantitative and qualitative results by adapting Force Scheme and t-SNE to *DimenFix* and fixing one dimension during the gradient descent process. We analyze *DimenFix* adaptations qualitatively and quantitatively, comparing them to the original versions of t-SNE and Force Scheme. For qualitative results, we use four benchmark datasets from the UC Irvine Machine Learning Repository [29]: **Iris** (4 dimensions, 150 instances), **Wine** (13 dimensions, 178 instances), **Breast Cancer** (30 dimensions, 569 instances), and **Segmentation** (19 dimensions, 2,310 instances), covering different scenarios where *DimenFix* can be useful.

In our first test, we compare the original Force Scheme (FS) and the *DimenFix* Force Scheme (FS-DF) adaptation layouts in Fig. 2. To generate such layouts, we randomly select one of the features of the original datasets to execute *DimenFix* (the feature values are assigned to the *y* axis). The features are: *sepal width* for **Iris**, *nonflavanoid_phenols* for **Wine**, *worst texture* for **Breast Cancer**, and *region-centroid-row* for **Segmentation**. For each layout, we show two figures side-by-side, the left colored by the class label and the right colored by the selected dataset features (where darker indicates larger values).

In the Force Scheme (FS) projections, it is possible to notice that the selected features do not smoothly vary across the layout from small to large values. The values are mixed without a clear trend. On the **Iris** projection, there is a variation pattern inside the two visible big groups, but interestingly, the tendencies for these groups do not align. The small group somewhat aligns with the *y* axis, while for the larger group, the alignment is not clear. So, although not perfect, some global trend can be observed. For the remaining three datasets, global tendencies are hardly visible. For the **Segmentation** dataset, most low and high values are somewhat associated with well-defined clusters, but in general, small and large values are mixed for the three datasets. When analyzing these layouts, not much can be said about the importance of such features to the produced layouts, and inferring the values of such features based on points' positions is not possible.

The Force Scheme *DimenFix* (FS-DF) results using *Clipping* and *Gaussian* modes attain much better results in terms of global tendencies regarding fixed features. Interestingly, for the **Breast Cancer** and *Iris* datasets, aligning one of the layout axes with a specific feature of the dataset does not have an evident negative impact on class separation. For the *Segmentation* and *Wine*, the separation is more impaired, with more class outliers presented on the produced layouts, although some separation is still present. From an analytical point of view, this allows for the execution of some interesting tasks; for instance, it is possible to verify the instances for which the selected feature has a negative impact in terms of group separation, and the ones that can be clearly differentiated based on such a feature.

An alternative, straightforward strategy to align 2D DR layouts with a given feature would be to create a one-dimensional layout to define one of the embedded axes and the other axis to be set to receive the selected feature values. The resulting layouts using this strategy for the Force Scheme (FS) technique are also shown in Fig. 2, identified as FS-1D. The results are indeed not bad; the *y*-axis represents the selected feature correctly, and the resulting layouts seem consistent, with groups still somewhat well separated — we later show that quantitatively, in terms of distance and neighborhood preservation, these layouts are not as good as the ones produced by *DimenFix*. However, an important drawback can be observed for the **Iris** dataset. Unlike the other datasets, where the selected features cover a range of different values, in the **Iris**, many instances have the same values and are defined in regular intervals (almost discrete), resulting in a "line" pattern with substantial overlap and poor use of the visual space.

The issue with discrete values is magnified when the values to be fixed are nominal with only a few different values, for instance, when data labels are used. Examples of using label information to fix one of the embedded axes are shown in Fig. 3. In these examples, we show the results of the adaptation of t-SNE to use *DimenFix*, as well as layouts using the original t-SNE technique, and layouts using the simple one-dimensional t-SNE plus class strategy (tSNE-1D) to exemplify one of the main benefits of *DimenFix*. In all *DimenFix* layouts, groups of instances are completely separated by labels in the produced layouts (in these

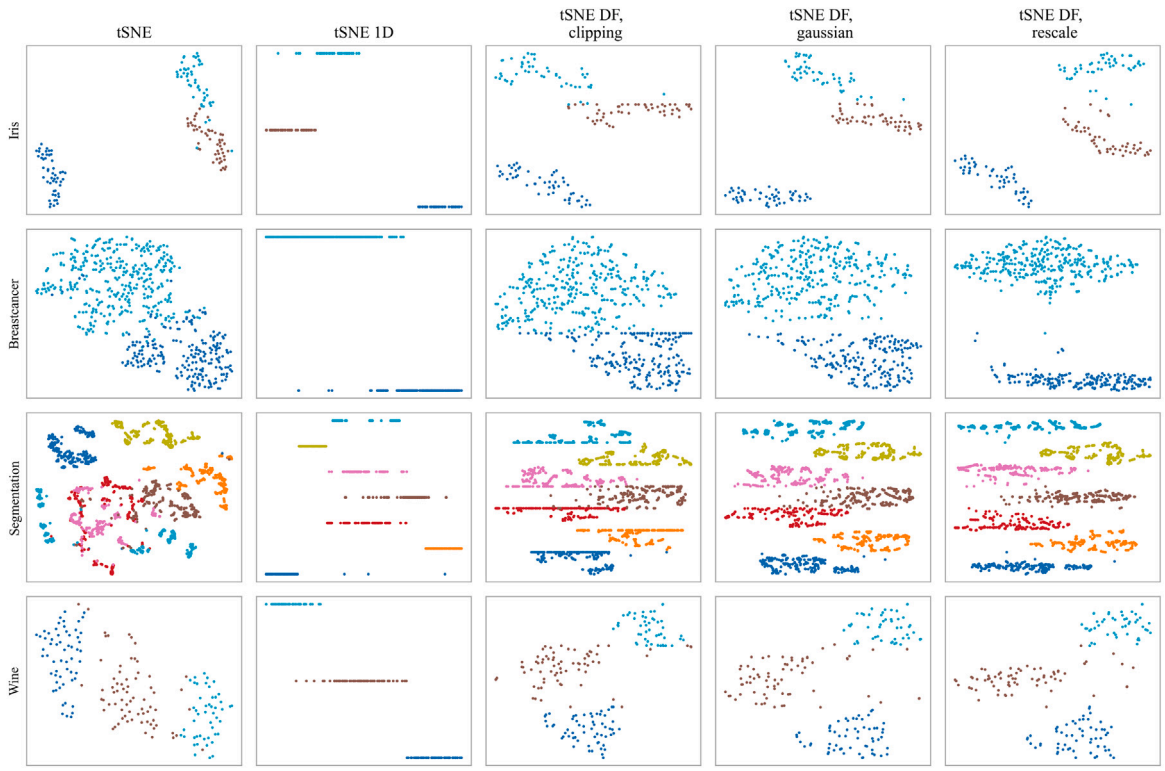


**Fig. 3.** t-SNE and t-SNE *DimenFix* adaptation layouts fixing the datasets labels (nominal) to the $y$-axis. *DimenFix* excels when compared with the straightforward idea of using a 1D t-SNE projection to define one axis and assigning the label values to the other. *DimenFix* allows the points to occupy the space between the classes in the $y$-axis, and the different pulling modes enable different results in terms of data instance groups representation.

examples, we set $\alpha = 1$), while data instances can still occupy the available space, resulting in layouts that make much better use of the available space to represent the relationships between data instances compared to the tSNE-1D strategy, where the same label instances occupy the same coordinate on the $y$-axis.

As expected, some of the layouts produced using the *DimenFix Clipping* pulling mode present apparent "lines" of instances over the boundaries between different label ranges. This results from the hard threshold employed in this strategy, where any instance moving outside its range is clipped to its boundary. This is especially true for datasets for which the original t-SNE cannot clearly separate the groups, in our examples, the *Breast Cancer* and *Segmentation* datasets. These "lines" are attenuated when *Gaussian* and *Rescale* pulling modes are employed. In these examples, we set the confidence interval of the *Gaussian* mode to $CI = 0.45$ to allow but minimize the overlap between different groups; only in extreme cases, instances with different labels overlap. This is completely avoided by using the *Rescale* mode. In addition, the shape of the groups of instances is better preserved in this mode. For instance, for the *Breast Cancer* dataset, some instances are dissimilar to instances in their groups. Using the *Rescale* mode, groups are separated, but such instances are positioned far apart from the instances of the group they belong to. This is not true for the *Gaussian* mode, where these outliers are hidden.

For all pulling modes, it is possible to set how much the values on the fixed axis can overlap. For any mode, if $\alpha = 0$ is used, the result is quite similar to the t-SNE-1D in Fig. 3, and all instances having the same label overlap on the fixed axis. For $0 > \alpha \geq 1$, the instances of different labels only overlap in the boundaries between neighbor ranges when $\alpha = 1$. For $\alpha > 1$, overlap starts to occur. Fig. 4 shows the results of varying $\alpha$ and the degree of overlap as it increases using the *Rescale* pulling mode. From $\alpha = 1.25$, overlap between class-outlier instances starts to occur, and when $\alpha = 2.0$, the two existing ranges fully overlap and the layout becomes very similar to the original t-SNE layout in Fig. 3. In this example, the classes are separated because the data allow

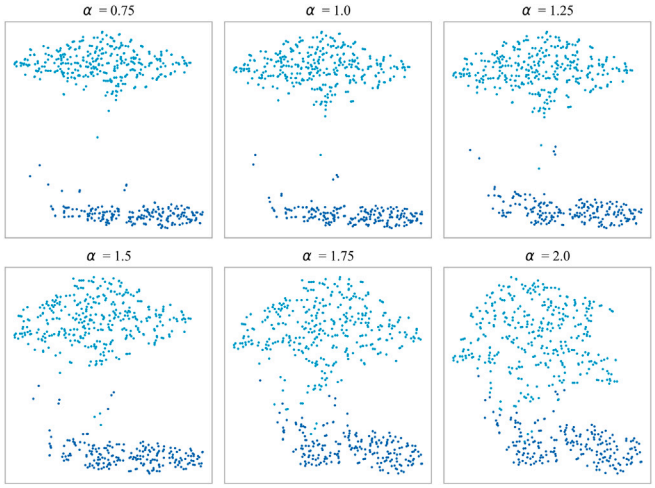


**Fig. 4.** Results of the t-SNE *DimenFix* adaptation for the *Breast Cancer* dataset with varying $\alpha$. As $\alpha$ increases, the separation between the two different groups of instances reduces, and the class-outlier instances start overlapping.

that, not because of *DimenFix*, although the classes are somewhat better separated than the original layout.

Lastly, we performed a quantitative analysis. Besides the four datasets used in the qualitative analysis, we added three datasets used in the survey [18] to cover more diverse scenarios, including the **imdb** (700 dimensions, 3250 instances), **epileptic** (178 dimensions, 5750 instances), and **spambase** (1024 dimensions, 3250 instances). In this analysis, we measure the pairwise distance preservation using Kruskal stress [30], the neighborhood preservation using trustworthiness/continuity [31], and class separation using distance consistency [32]. For

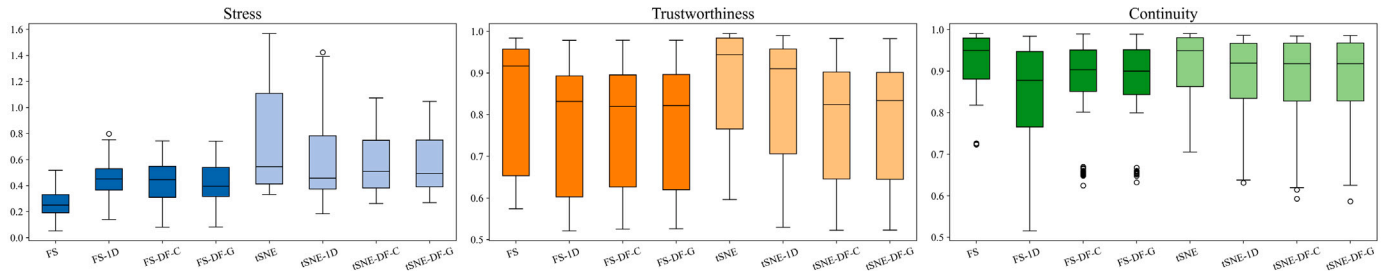


**Fig. 5.** Comparison of 1D and 2D Force Scheme and t-SNE with different *DimenFix* modes (clipping (C), Gaussian (G), and rescale(R)) for fixing the second axis to an ordinal value. *DimenFix* allows for feature value preservation with only a small loss of quality of the attained layouts in terms of distance and neighborhood preservation.

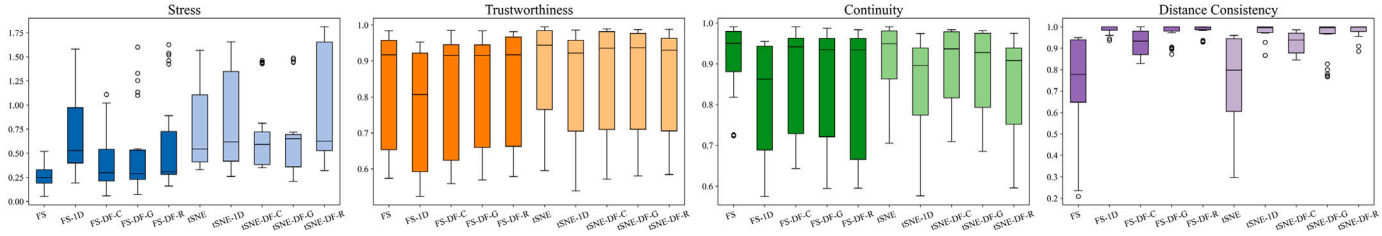


**Fig. 6.** Comparison of 1D and 2D Force Scheme and t-SNE with different *DimenFix* modes (clipping (C), Gaussian (G), and rescale(R)) for fixing the second axis to class labels. In general, as can be expected, distance and neighborhood preservation drop when nominal values are fixed, but the decrease is not too substantial, and *DimenFix* is consistently better than the simple 1D approximations.

trustworthiness, continuity, and distance consistency, the larger the better. For stress, it is the opposite.

Boxplots summarizing executions fixing dataset features are shown in Fig. 5. Each dataset is embedded 20 times, starting with random configurations and randomly selecting the fixed feature. Boxplots summarizing executions fixing the dataset labels are shown in Fig. 6. In this case, each dataset is embedded five times with random initial configurations. We show results of the original Force Scheme (FS) and t-SNE techniques (tSNE), the 1D Force Scheme (FS-1D) and 1D t-SNE (tSNE-1D) approximations, and the *DimenFix* adaptations of the Force Scheme (FD-DF) and t-SNE (tSNE-DF) considering the different pulling modes: clipping (C), Gaussian (G), and rescale(R). In all experiments, we set ForceScheme's maximum number of iterations to $max = 100$. For t-SNE, such a parameter does not apply, and we used the scikit-learn standard parameters, only setting perplexity to 15 and initialization to random. For the *DimenFix* parameters, we set $\alpha = 1$ and the *number of iterations to pull points* to $k = 10$. There are no general rules to set $\alpha$, since it depends on the users' preference. The number of iterations to pull ($k$) should be set so that the pulling and the GD updates work "together" to define the final layout. Through experimental results, we find $k = 10$ a good compromise between the distortion level per iteration introduced by a pulling and the increased cost of executing it. However, this can differ for other techniques.

When fixing dataset ordinal features (Fig. 5), as expected, the original Force Scheme achieved the best results for stress and the original t-SNE for trustworthiness/continuity, since two dimensions are fully used to approximate the original distances (and neighborhoods) and those techniques are known for their quality in those aspects [18]. Nevertheless, in general, the results when fixing features are not much worse for both the *DimenFix* adaptations and the 1D approximations (sometimes unexpectedly better), indicating that the resulting layouts are somewhat similar to those generated by the original techniques with respect to distance and neighborhood preservation. Also, Force Scheme *DimenFix* with Gaussian pulling(FS-DF-G) stress and continuity results are better than the 1D approximation (FS-1D and tSNE-1D), while trustworthiness attains similar results. The opposite happens for t-SNE (tSNE-DF), stress and trustworthiness are better for the 1D approximation, while continuity attains similar values. This indicates that the use of information from both dimensions in the gradient descent process is beneficial when adapting the Force Scheme, a global technique, while this is not observed for the t-SNE adaptation, a local technique. Nevertheless, the difference is marginal, and either of the two types of adaptations could be used interchangeably.

However, when nominal labels are fixed (Fig. 6), overall preservation decreases more substantially (stress, truthworthiness, and continuity). This is expected given the labels' discrete nature and the non-ordinal relationship between the labels' values in any of these datasets (label 1 is not larger than label 0). This has a more pronounced impact on the 1D approximations, with *DimenFix* presenting better results in at least one of the pulling modes, indicating that the proposed nominal rotation and ordering strategies address this ordering issue, especially for the Force Scheme adaptations. Not surprisingly, *DimenFix* presents better class separation (distance consistency) when compared with the original Force Scheme and t-SNE techniques since labels are not used by such techniques. Notice that the 1D approximations also present very good results in terms of distance consistency. Distance consistency measures the percentage of points that are closer to the centroid of their own class than to the centroids of the other classes. Since the 1D approximations position the points on lines (see Fig. 3), distance consistency can be high, but the other metrics show that the layouts are distorted. This further indicates that taking the fixed feature/class into consideration in the optimization process results in more precise projections that can convey relationships between features or classes and the point positions. This is not usually possible using the original Force Scheme and t-SNE techniques.

## 5. Conclusion and future work

This paper presents a novel meta-Dimensionality Reduction (DR) strategy, *DimenFix*, built upon any gradient-descent-based DR method. Unlike normal DR methods, *DimenFix* allows users to fix a feature/class of the dataset (or any external data, for example, labels) to an axis in the created layout without affecting its quality (too strongly) so that the spatialization reflects the fixed values. With this hybrid strategy joining DR and scatter plot capabilities, different analytical goals can be achieved, for instance, (a) better preserving a particular feature of the dataset while simultaneously reducing the dimensionality and (b) understanding the data distribution with respect to a specific feature specified by the user.

Despite promising results, claims about high-quality distance and neighborhood preservation are constrained by the limited set of tests we executed. Although we tested on datasets with different dimensionalities and sizes, we do not claim any guarantee of high-quality preservation holding to other datasets. Thus, whenever executing *DimenFix* adaptations, the resulting layout should be validated using metrics of interest (e.g. stress and trustworthiness). However, there is good evidence that *DimenFix* will be consistently better than simple 1D approximations when nominal values are fixed, e.g., dataset labels. As such, if a task involves the analysis of a particular feature, or if it includes external information, for instance, labels, time, or a ranking, *DimenFix* should be considered as an option.

The adaptation of other gradient descent DR techniques to use *DimenFix*, such as UMAP [17] should be straightforward for most existing techniques. *DimenFix* is minimally intrusive as it only requires access to the projection coordinates after an update of the GD iteration ($\overline{Y}$ in line 6 of Algorithm 1). The only important consideration is that a *DimenFix* pull needs to be the last step to be executed in a DR process. Modifications in the projection coordinates after a pull, such as any required post-processing, may negatively influence the results. In case any technique needs such a pre-processing, we do not recommend the use of *DimenFix*, or we suggest executing a *DimenFix* pull after such pre-processing.

*DimenFix* could benefit from some future work. The rotation for nominal values we propose is simple and effective; however, we are aligning the layout considering only the two most distant groups of points. A different strategy involving an optimization to consider all groups of instances could potentially result in more precise layouts. The challenge is to create a strategy that is not too computationally expensive. Further, more sophisticated pulling strategies could be designed. In the proposed strategies, *Clipping*, *Gaussian*, and *Rescale*, the layout points move independently and parallel to the fixed axis, since this results in the smallest overall displacement. However, a better distance or neighborhood preservation could be achieved if points move together, allowing diagonal movements. Once again, the challenge is to define a strategy that does not affect the overall computational complexity of the underlying DR technique. In this paper, our focus was to keep the elements involved simple but effective, and our complexity is $O(N)$, where $N$ is the number of data instances for any of the steps proposed (range and position calculation, rotation, ordering, and pulling).

Finally, the way we present *DimenFix* allows setting the dimensionality of the projection $Y$ to any number (considering the underlying DR strategy also supports that), for instance, allowing the creation of 3D projections without any modification in the proposed strategies. Although 3D projections require more sophisticated interactive environments and are prone to occluding problems, that may be beneficial for *DimenFix* layouts since the gradient descent process recovers the 2 degrees of freedom as in 2D projections, potentially leading to better distance and neighborhood preservation. Also, this opens the possibility of designing a pulling strategy that results in one of the orthogonal views of the 3D projection being the same as the 2D projection, so the 3D representation would contribute to a more informative exploratory process. Further investigation of 3D *DimenFix* layouts is left for future work.

## CRediT authorship contribution statement

**Zixuan Han:** Software, Methodology, Investigation, Conceptualization. **Diede van der Hoorn:** Writing – review & editing, Validation, Software. **Thomas Höllt:** Writing – review & editing, Supervision, Investigation, Conceptualization. **Qiaodan Luo:** Software, Methodology, Conceptualization. **Leonardo Christino:** Supervision, Software, Conceptualization. **Evangelos Milios:** Writing – review & editing, Supervision. **Fernando V. Paulovich:** Writing – original draft, Supervision, Software, Conceptualization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Code availability

The code of this study is available from https://github.com/fpaulovich/dimensionality-reduction/.

## References


[1] Viegas RG, Martins IBS, Sanches MN, Oliveira Junior AB, Camargo JBd, Paulovich FV, et al. ELViM: Exploring biomolecular energy landscapes through multidimensional visualization. J Chem Inf Model 2024;64(8):3443–50. http://dx.doi.org/10.1021/acs.jcim.4c00034.

[2] Oliveira, Jr. AB, Fatore FM, Paulovich FV, Oliveira , Jr. ON, Leite VBP. Visualization of protein folding funnels in lattice models. PLoS One 2014;9(7):1–9. http://dx.doi.org/10.1371/journal.pone.0100861.

[3] Oliveira, Jr. ON, Pavinatto FJ, Constantino CJL, Paulovich FV, de Oliveira MCF. Information visualization to enhance sensitivity and selectivity in biosensing. Biointerphases 2012;7(1):53. http://dx.doi.org/10.1007/s13758-012-0053-7.

[4] Volpati D, Aoki PHB, Dantas CAR, Paulovich FV, de Oliveira MCF, Oliveira, Jr. ON, et al. Toward the optimization of an e-tongue system using information visualization: A case study with perylene tetracarboxylic derivative films in the sensing units. Langmuir 2012;28(1):1029–40. http://dx.doi.org/10.1021/la203641a.

[5] Perinoto ÂC, Maki RM, Colhone MC, Santos FR, Migliaccio V, Daghastanli KR, et al. Biosensors for efficient diagnosis of leishmaniasis: Innovations in bioanalytics for a neglected disease. Anal Chem 2010;82(23):9763–8. http://dx.doi.org/10.1021/ac101920t.

[6] Kobak D, Berens P. The art of using t-SNE for single-cell transcriptomics. Nat Commun 2019;10(1):5416. http://dx.doi.org/10.1038/s41467-019-13056-x.

[7] Aksamit N, Hou J, Li Y, Ombuki-Berman B. Integrating transformers and many-objective optimization for drug design. BMC Bioinformatics 2024;25(1):208. http://dx.doi.org/10.1186/s12859-024-05822-6.

[8] Sacha D, Zhang L, Sedlmair M, Lee JA, Peltonen J, Weiskopf D, et al. Visual interaction with dimensionality reduction: A structured literature analysis. IEEE Trans Vis Comput Graphics 2017;23(1):241–50. http://dx.doi.org/10.1109/TVCG.2016.2598495.

[9] Coimbra DB, Martins RM, Neves TT, Telea AC, Paulovich FV. Explaining three-dimensional dimensionality reduction plots. Inf Vis 2016;15(2):154–72. http://dx.doi.org/10.1177/1473871615600010.

[10] Sohns JT, Schmitt M, Jirasek F, Hasse H, Leitte H. Attribute-based explanation of non-linear embeddings of high-dimensional data. IEEE Trans Vis Comput Graphics 2022;28(1):540–50. http://dx.doi.org/10.1109/TVCG.2021.3114870.

[11] Marcílio-Jr WE, Eler DM, Garcia RE. Contrastive analysis for scatterplot-based representations of dimensionality reduction. Comput Graph 2021.

[12] Marcílio-Jr WE, Eler DM. Explaining dimensionality reduction results using Shapley values. Expert Syst Appl 2021;178:115020. http://dx.doi.org/10.1016/j.eswa.2021.115020.

[13] Tian Z, Zhai X, van Driel D, van Steenpaal G, Espadoto M, Telea A. Using multiple attribute-based explanations of multidimensional projections to explore high-dimensional data. Comput Graph 2021.

[14] Thijssen J, Tian Z, Telea A. Scaling Up the Explanation of Multidimensional Projections. In: Angelini M, El-Assady M, editors. EuroVis workshop on visual analytics. The Eurographics Association; 2023, p. 61–6. http://dx.doi.org/10.2312/eurova.20231098.

[15] Tejada E, Minghim R, Nonato LG. On improved projection techniques to support visual exploration of multidimensional data sets. Inf Vis 2003;2(4):218–31. http://dx.doi.org/10.1057/palgrave.ivs.9500054.

[16] van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res 2008;9(86):2579–605.

[17] Healy J, McInnes L. Uniform manifold approximation and projection. Nat Rev Methods Prim 2024;4(1):82. http://dx.doi.org/10.1038/s43586-024-00363-x.

[18] Espadoto M, Martins RM, Kerren A, Hirata NST, Telea AC. Toward a quantitative survey of dimension reduction techniques. IEEE Trans Vis Comput Graphics 2021;27(3):2153–73. http://dx.doi.org/10.1109/TVCG.2019.2944182.

[19] Nonato LG, Aupetit M. Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment. IEEE Trans Vis Comput Graphics 2019;25(8):2650–73. http://dx.doi.org/10.1109/TVCG.2018.2846735.

[20] Torgerson WS. Multidimensional scaling: I. Theory and method. Psychometrika 1952;17:401–19. http://dx.doi.org/10.1007/BF02288916.

[21] Paulovich FV, Silva CT, Nonato LG. Two-phase mapping for projecting massive data sets. IEEE Trans Vis Comput Graphics 2010;16(6):1281–90. http://dx.doi.org/10.1109/TVCG.2010.207.

[22] Paulovich FV, Nonato LG, Minghim R, Levkowitz H. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. IEEE Trans Vis Comput Graphics 2008;14(3):564–75. http://dx.doi.org/10.1109/TVCG.2007.70443.

[23] Heulot N, Fekete JD, Aupetit M. Visualizing dimensionality reduction artifacts: An evaluation. 2017, URL https://arxiv.org/abs/1705.05283, arXiv:1705.05283.

[24] Bentley C, Ward M. Animating multidimensional scaling to visualize N-dimensional data sets. In: Proceedings IEEE symposium on information visualization '96. 1996, p. 72–3. http://dx.doi.org/10.1109/INFVIS.1996.559223.

[25] Schreck T, von Landesberger T, Bremm S. Techniques for precision-based visual analysis of projected data. Inf Vis 2010;9(3):181–93. http://dx.doi.org/10.1057/ivs.2010.2.

[26] Martins RM, Coimbra DB, Minghim R, Telea AC. Visual analysis of dimensionality reduction quality for parameterized projections. Comput Graph 2014.

[27] Sedlmair M, Aupetit M. Data-driven evaluation of visual quality measures. Comput Graph Forum 2015;34(3):201–10. http://dx.doi.org/10.1111/cgf.12632.

[28] Silva RROd, Rauber PE, Martins RM, Minghim R, Telea AC. Attribute-based visual explanation of multidimensional projections. In: Bertini E, Roberts JC, editors. EuroVis workshop on visual analytics. The Eurographics Association; 2015, p. 31–5. http://dx.doi.org/10.2312/eurova.20151100.

[29] Kelly M, Longjohn R, Nottingham K. The UCI Machine Learning Repository. 2024, URL https://archive.ics.uci.edu.

[30] Kruskal JB. Nonmetric multidimensional scaling: a numerical method. Psychometrika 1964;29(2):115–29. http://dx.doi.org/10.1007/BF02289694.

[31] Venna J, Peltonen J, Nybo K, Aidos H, Kaski S. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. J Mach Learn Res 2010;11(2).

[32] Sips M, Neubert B, Lewis JP, Hanrahan P. Selecting good views of high-dimensional data using class consistency. Comput Graph Forum 2009;28(3):831–8. http://dx.doi.org/10.1111/j.1467-8659.2009.01467.x.