Information Visualization
1–19

DOI: 10.1177/1473871615600010
ivi.sagepub.com
SAGE

Explaining three-dimensional dimensionality reduction plots

Danilo B Coimbra[1], Rafael M Martins[1], Tácito TAT Neves[1], Alexandru C Telea[2] and Fernando V Paulovich[1]

Abstract

Understanding three-dimensional projections created by dimensionality reduction from high-variate datasets is very challenging. In particular, classical three-dimensional scatterplots used to display such projections do not explicitly show the relations between the projected points, the viewpoint used to visualize the projection, and the original data variables. To explore and explain such relations, we propose a set of interactive visualization techniques. First, we adapt and enhance biplots to show the data variables in the projected three-dimensional space. Next, we use a set of interactive bar chart legends to show variables that are visible from a given viewpoint and also assist users to select an optimal viewpoint to examine a desired set of variables. Finally, we propose an interactive viewpoint legend that provides an overview of the information visible in a given three-dimensional projection from all possible viewpoints. Our techniques are simple to implement and can be applied to any dimensionality reduction technique. We demonstrate our techniques on the exploration of several real-world high-dimensional datasets.

Keywords

Dimensionality reduction, scatterplots, multivariate visualization, explanatory visualization

Introduction

Dimensionality reduction (DR) techniques are an important part of visual analytics solutions. DR techniques map, or project, datasets having tens or even hundreds of variables into a low-dimensional space (two-dimensional (2D) or three-dimensional (3D)), so that distance and/or neighborhood relations between data points—the so-called data structure—are preserved. Projected results can be next visualized by techniques such as scatterplots[1] and parallel coordinates.[2] DR methods have been used for the analysis of text documents,[3–7] multimedia,[8,9] text mining,[10,11] vector fields,[12] and biomedical data.[13–15]

DR techniques have become very robust, precise, computationally scalable, and easy to apply. While 2D projections require less effort to explore,[16–18] 3D projections preserve better the original high-dimensional data structure.[6,19–21] However, 3D projections output a 3D point cloud, typically shown as a scatterplot, whose interpretation is far from simple.[1] As users rotate the scatterplot to find a suitable viewpoint, several questions arise, such as how much of the original data structure has the projection preserved? What is the meaning of the 3D directions along which scatterplot points are spread in terms of original variables' values and/or correlations? What are good viewpoints to look at the scatterplot from, given a set of questions on these variables?

We propose a set of interactive explanatory visualization techniques to help users answer the above questions for 3D DR projections. Our techniques work as

[1]University of São Paulo, São Carlos, Brazil
[2]University of Groningen, Groningen, The Netherlands

Corresponding author:
Danilo B Coimbra, University of São Paulo, Avenida Trabalhador São-carlense, 400 – Centro, São Carlos 13566-590, Brazil.
Email: danilobc@icmc.usp.br

add-ons to any DR technique, that is, do not depend on technical aspects of the DR algorithm being used. We keep their visual design simple, so that learning to use them requires limited effort. We integrate our techniques with classical 3D scatterplot views, so that they can be readily used to assist typical projection–exploration scenarios, or in other words, *explain* the projection. Specifically, we show how our techniques can aid detecting global correlations of variables, by suitably changing the viewpoint via 3D trackball-like rotations and by explaining which variables are best visible from a given viewpoint and which are not, due to occlusion or screen-space projection. We illustrate our visualization techniques by applying them to several data-exploration scenarios involving real-world multidimensional datasets and a set of recent DR projection algorithms.

The structure of this article is as follows. Section "Related work" presents related work on the computation and interactive exploration of DR projections and also outlines several goals supported by such exploration. Section "Explanatory visualizations" introduces our explanatory visualizations via a simple dataset. Section "Applications" illustrates how our visualizations can answer several questions on 3D scatterplots created by several DR techniques from real-world datasets. Section "Discussion" discusses our techniques. Finally, section "Conclusion" concludes the article.

Related work

DR

Given a dataset $D^n = \{\mathbf{p}_i \in \mathbb{R}^n\}_{1 \leqslant i \leqslant N}$ of N n-dimensional points, we model DR as a function

$$f: \quad \mathbb{R}^n \times \mathbb{P} \rightarrow \mathbb{R}^m \tag{1}$$

which maps each point $\mathbf{p}_i \in D^n$ to a point $\mathbf{q}_i \in D^m \subset \mathbb{R}^m$. Here, n is typically large (tens up to hundreds of dimensions, or variables), and m is typically 2 or 3. $\mathbb{P}$ denotes the parameter space of f, that is, the settings that control the projection algorithm. f is designed to keep the so-called *structure* of the data as similar as possible in $\mathbb{R}^n$ and $\mathbb{R}^m$. One way for this is to let f minimize the normalized stress function

$$\sigma = \frac{\sum_{1 \leqslant i \leqslant N, 1 \leqslant j \leqslant N} \left(d^n(\mathbf{p}_i, \mathbf{p}_j) - d^m(\mathbf{q}_i, \mathbf{q}_j)\right)^2}{\sum_{1 \leqslant i \leqslant N, 1 \leqslant j \leqslant N} \left(d^n(\mathbf{p}_i, \mathbf{p}_j)\right)^2} \tag{2}$$

where $d^n: \quad \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+$ and $d^m: \quad \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}^+$ are the distance metrics for D^n and D^m, respectively. Alternatively, f can optimize for having the k-nearest neighbors for a point $\mathbf{q}_i \in D^m$ be the same as the k-nearest neighbors of $\mathbf{p}_i \in D^n$, for an application-dependent k.

Many DR methods are a special case of a wider class of techniques called multidimensional scaling (MDS). MDS methods compute f using only pairwise point distances. This avoids accessing the full dataset D^n. Yet, computing distances creates additional costs ($O(N^2)$ for N data points). The part-linear multidimensional projection (PLMP) method avoids this by using distances only for a small set of *sample* (representative) points in D^n and using the nD coordinates for the remaining points.[14]

DR methods can be classified by the techniques used to compute f:[14] *spectral decomposition* techniques project points along the eigenvectors having the largest eigenvalues of the pointwise distance matrix.[22] Local linear embedding (LLE)[23] and isometric feature mapping (ISOMAP)[24,25] use numerical methods tailored to solve sparse eigen problems. Landmark MDS[26] and Pivot MDS[27] book further speed-ups by using classical MDS on a small set of sample points and projecting remaining points by local interpolation. Fastmap achieves linear complexity in the input point count at the cost of a less well-minimized stress σ.[8] *Nonlinear optimization* methods iteratively search the parameter space P to minimize the stress σ.[28,29] Besides naive gradient descent, multigrid numerical solvers are used to speed searching.[30] Pekalska et al.[31] propose a speed-up that projects a sample subset (by gradient descent) and fits remaining points by local interpolation. Force-based methods are a special class of nonlinear methods, often used in graph drawing.[32] Chalmers[33] speeds these up by using the sample subset idea outlined earlier. Further speed-ups are achieved by multilevel solvers and GPU techniques[34,35] and by recursively selecting samples via a multilevel approach.[36] Tejada et al.[37] use a heuristic to embed instances by a force-based relaxation technique. The least squares projection (LSP) positions the sample subset by a force-based scheme and fits remaining points by Laplacian smoothing.[6] The local affine multidimensional projection (LAMP) uses a sample subset to locally construct affine projections and allows users to interactively place such points to optimize the overall projection layout.[9] Local projection methods preserve distances better for high-dimensional spaces (high n values), but are more computationally demanding and more complex to implement.[9,38] Global projection methods preserve distances better for low-dimensional spaces (low n) values and are also relatively faster and simpler to implement. For more details on these and other DR techniques, we refer to recent surveys in Sorzano et al.[39] and Van der Maaten et al.[40]

Explaining projections

Interpreting DR scatterplots is not easy. Refining the questions in section "Introduction," we identify the following goals which we aim to address:

1. Assign a meaning to the m dimensions of the mD projection space in relation to the original n variables.
2. Assign a meaning to the inter-point distances in $\mathbb{R}^m$ in relation to the corresponding distances in $\mathbb{R}^n$.
3. Find a suitable viewpoint (for 3D projections) that best supports answering specific questions.
4. Compare the quality of projections for dimensions $m \in \{2, 3\}$ from the perspective of several given tasks.

Goal 1 can be addressed by *biplots* and their variations.[41,42] Biplots are the multivariate analog of scatterplots. Instead of using the scatterplot idea of plotting observations along two orthogonal (Cartesian) axes mapping two variables, biplots approximate the multivariate distribution of a high-dimensional dataset in a few dimensions, typically 2 or 3, by superimposing representations of variable values on representations of the observations themselves. As such, they offer the possibility to easily see relationships between (1) individual observations and (2) observations and their variable values.[43] Graphically, biplots can be seen as a scatterplot generalization, in the sense that they have as many axes as there are variables, and these axes can take any orientation in the display. Biplot axes support goal (2) above by showing which are the directions of maximal variation in the original n variables in the m-dimensional projection space.

Biplots and their axes are usually constructed as follows. Consider the $N \times n$ matrix $\mathbf{D} = (\mathbf{p}_i)_{1 \leqslant i \leqslant N}$. If $\mathbf{D}$ has rank r, it can be rewritten by singular value decomposition (SVD) as

$$\mathbf{D} = \mathbf{U}\Delta\mathbf{V}^{\mathrm{T}} \tag{3}$$

where $\mathbf{U}$ is an $N \times r$ matrix, Δ is a $r \times r$ diagonal matrix of eigenvalues $\alpha_1 > \cdots > \alpha_r > 0$, and $\mathbf{V}$ is an $n \times r$ matrix. Here, $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Denoting $\mathbf{F} = \mathbf{U}\Delta$, we have $\mathbf{D} = \mathbf{F}\mathbf{V}^{\mathrm{T}}$. The columns of $\mathbf{V}^{\mathrm{T}}$ define the biplot *axes*. The rows of matrix $\mathbf{F}$ define the *projections* of our data points $\mathbf{p}_i$ onto these axes. If $r \leqslant 3$, we can directly visualize the biplot by drawing projections as a point cloud and biplot axes as vector glyphs (oriented straight lines), respectively. If $r > 3$, we can approximate $\mathbf{D}$ by using in equation (3) only the first $m < r$ columns of $\mathbf{U}$ and $\mathbf{V}$. Then $\mathbf{F}$ gives the m-dimensional projections $\mathbf{q}_i$ of $\mathbf{p}_i$ along the eigenvectors corresponding to the m largest eigenvalues $\alpha_1, \ldots, \alpha_m$ of $\mathbf{D}$. Using eigenvectors as biplot axes, however, does not convey much insight, as eigenvectors usually do not relate one-to-one to the original variables in D^n. A better solution is to construct n biplot axes by projecting, via equation (3), the n unit vectors in $\mathbf{R}^n$. These vectors show the direction of maximal variation in the resulting m-dimensional projection of our n variables.[42,44]

A different approach to Goal 1 is given in Broeksema et al.[45] Here, an nD categorical dataset is projected to $m = 2$ dimensions by SVD. Instead of drawing n biplot axes, the contributions to the screen x- and y-axes of all original n dimensions are shown. These contributions, also called *loadings*,[42,44] are the projections of the nD unit vectors (via equation (3)) on the two eigenvectors that determine the projection. The x- and y-axes are annotated with two n-element bar charts, where the height of each bar shows the contribution of a given variable to the respective axis. A third bar chart shows the contributions of all n variables to all eigenvectors *not* used to construct the DR projection. This shows the amount of data variance not captured by the 2D projection. A similar visualization of loadings is shown in Oeltze et al.[13]

Goal 2, that is, assigning a meaning to the inter-point distances in $\mathbb{R}^m$ in relation to the corresponding distances in $\mathbb{R}^n$, is addressed by aggregated quality metrics such as stress (equation (2)), correlation,[46] neighborhood-preservation plots,[6] shape-based metrics,[47] and perceptual user studies.[48] While showing the *overall* quality of a projection, aggregate metrics do not show *local* projection errors. 2D distance scatterplots can show the correlation of D^n with D^m,[9] but do not show projection problems for *any* point i versus *all* points $j \neq i$. To improve this, Schreck et al.[49] compute, for each data point $\mathbf{p}_i$, the projection precision score (*pps*) defined as the normalized distance between the two k-dimensional vectors containing the Euclidean distances between $\mathbf{p}_i$ and its k-nearest neighbors in $\mathbb{R}^n$, respectively $\mathbb{R}^2$. Showing *pps* via a colormap helps finding areas where neighborhoods are not preserved. Aupetit[50] proposed several projection-quality metrics for DR techniques: Point stretching and compression metrics show, for each $\mathbf{p}_i$, the aggregated increase and, respectively, the decrease, of distances of $\mathbf{p}_i$'s projection $\mathbf{q}_i$ to other projections $\mathbf{q}_{j \neq i} \in D^m$ versus distances of $\mathbf{p}_i$ to all other points $\mathbf{p}_{j \neq i} \in D^2$. Segment stretching and compression show the variation in distances of close point-pairs (i, j) between $\mathbb{R}^n$ and $\mathbb{R}^2$. For a selected point i, the proximity metric maps distances in $\mathbb{R}^n$ from $\mathbf{p}_i$ to all other points $\mathbf{p}_{j \neq i}$ to corresponding points $\mathbf{q}_i \in \mathbb{R}^2$, and thereby shows how (and where) the projection may have distorted the data structure. An overview of quality metrics for multivariate data visualization is given in Bertini et al.[51]

Goals 1 and 2 are also addressed jointly by other tools. The early VIBE system allows users to freely place in 2D space several so-called points of interest (POIs), each representing a sample of the *n*D space under study.[5] Points in this space represent documents along *n* dimensions encoding term frequencies. Actual documents are placed in the same 2D space so as to reflect their relative similarities with the given POIs. Conceptually, this can be seen as projecting both documents and POIs (variable values) from *n*D to 2D. However, this approach requires the user to manually create relevant POIs (samples of the *n*D space) and also place them suitably in 2D. ForceSPIRE, a document-exploration system, uses a force-based layout to construct a 2D projection of a set of documents represented as *n*D term vectors.[4] By dragging, pinning, and annotating documents, users can incrementally assign higher-level *semantics* to 2D inter-document distances. The "dust & magnets" technique extends the exploration power of ForceSPIRE and VIBE by allowing users to interactively drag magnets to discover how data points (dust) are attracted toward them in an animated fashion.[7] While we also use interaction to explain a projection, like Endert et al.,[4] Olsen et al.,[5] and Yi et al.[7] our focus is to explain projection-space distances in terms of the original *n*D variables, rather than showing similarities of projected points with a user-selected set of variable values or extracting higher-level semantics from variable values. As such, we will not modify the projection, as we consider it to be our "ground truth" and also give a key role to the *n*D variables in our explanation.

Goal 3, that is, finding a suitable viewpoint (for 3D projections) that best supports answering specific questions, can be addressed by multiple views, such as three 2D views linked with a 3D scatterplot by interactive selection,[52] or interaction and animation, for example, the scatterplot matrix. "Rolling the dice" (RTD) adds interactivity to improve navigation, 3D animated transitions to explore the visual space, and swapping the scatterplot-matrix axes to show variable correlations and disparities.[1] This idea was extended in Sanftmann and Weiskopf[53] by linking a 3D scatterplot with a 3D scatterplot matrix, improving navigation by using three axes and using one or two axes during visual transitions. A similar idea was used by Hurter et al.[54] to link 3D and 2D scatterplots. Claessen and Van Wijk[55] extend axis movement for scatterplot navigation, to allow users to interactively draw, place, and link axes on a canvas, thereby creating a continuous combination space of 2D scatterplots, scatterplot matrices, and parallel coordinates.

Goal 4, that is, comparing 2D versus 3D DR projections, to find which is more suitable for a specific context (and why), is still an open subject.[56] Several authors argue that 2D DR plots are better for visualizing text documents,[16,18] and that 2D navigation is easier than its 3D counterpart.[18] For the specific task of cluster separation, Sedlmair et al.[17] argue that 2D DR plots are found to be as good as (interactive) 3D DR plots. 2D DR plots were also found better for search tasks[57] and for tasks involving distance assessment and spatial arrangements.[58] On the other hand, Jolliffe[19] argues that 3D projections are needed to "encode a realistic picture of what the data look like" when the intrinsic data dimension is 3 or higher. Dang et al.[59] show how 3D glyph stacking can overcome color coding problems in 2D plots. Additional cues such as illumination and depth are proposed in support of using 3D scatterplots.[60] Sanftmann and Weiskopf[53] argue that high-point densities in scatterplots are better handled by 3D scatterplots. Chan et al.[61] argue that 3D projections decrease information loss by allowing better discrimination between data elements. A discussion of contexts where 3D DR projections are preferable to 2D ones is given in Sanftmann and Weiskopf[62] (section 2.3). Poco et al.[21] compared 2D and 3D DR projections using LSP[6] both quantitatively (by stress metrics) and qualitatively (by controlled user studies). The quantitative comparisons showed a higher accuracy of 3D projections; the user studies showed that when augmented by suitable interaction tools, 3D projections were superior to 2D projections in terms of both confidence and satisfaction and argued for the further development of 3D interactive exploration tools.

Summarizing the above, with needed brevity, we argue that (a) 2D DR plots are generally found more effective for the specific tasks of cluster separation and searching and require less interaction; while 3D DR plots preserve distances better, but loose appeal due to navigation, orientation, and occlusion problems. As such, we argue that our goal of designing effective interactive exploration tools for 3D DR projections, which keep the benefit of higher 3D projection accuracy as compared to 2D projections, but decrease 3D interpretation costs, is worth investigating.

Explanatory visualizations

We next detail our interactive visualizations that support the explanatory goals in section "Explaining projections." As running example, we use a dataset containing 2814 points, each representing the abstract of a scientific article (dataset *ALL* in Paiva et al.[63]). From the abstracts, a nine-dimensional feature space was created by removing stop-words and using stemming. Feature coordinates were computed by the term-frequency-inverse-document-frequency count.[64]

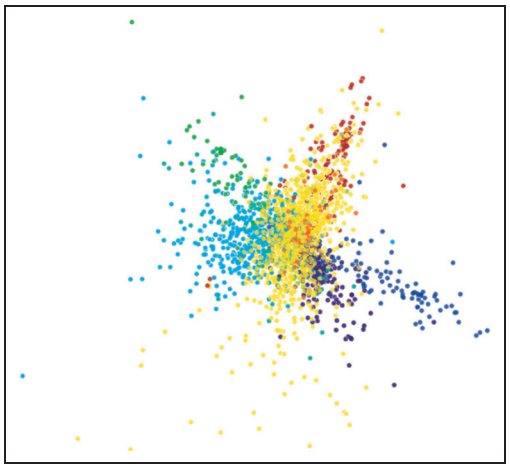

Figure 1. Document dataset shown by a 3D scatterplot.

From this dataset, a 3D projection was created using LAMP.[9] A 10th attribute, not used in the projection, indicates the class of each document, established manually based on the perceived topic of each document.

Figure 1 shows the 3D projection using a scatterplot, with points colored by their class attribute. Apart from seeing a few separated point clusters, which seem to capture the class attribute, this image does not tell us more: we do not know how variable values vary along the 3D space, or whether they correlate with the clusters or with each other, or how to choose a good viewpoint to examine the dataset. We next show how to answer such questions.

Enhanced biplot axes

Standard biplots project the n variables into biplot axes in the low-dimensional mD space using SVD (equation (3)), as described in section "Explaining projections." This has several problems. First, this assumes that DR is done using a uniform *and* linear transformation. This is not true for nonlinear DR techniques or techniques based on different local projection schemes ("Related work"). Second, this assumes that we know the internals of the DR method, such as the SVD matrices $\mathbf{U}$, Δ, and $\mathbf{V}$ (section "Explaining projections). Finally, such biplots cannot show the direction and (nonlinear) scaling of the n variables.

We address these issues as follows. For each nD variable i, we create a set of $S = 100$ sample points $\mathbf{p}^i_{1 \leqslant j \leqslant S}$, spread uniformly between the minimum and maximum of variable i in D^n; for the other variables $k \neq i$, $\mathbf{p}^k_j$ take values equal to the average of variable k in D^n. Next, we use the DR projection f (equation (1)) as a *black box* to project the points $\mathbf{p}^i_j$ to $\mathbf{q}^i_j \in \mathbb{R}^m$ and draw a curve (biplot axis) $c_i = \{\mathbf{q}^i_j\}$ to connect all projected points. Figure 2(a) shows this. We see how and where the nD axes get mapped in the 3D space. The lengths and bends of the curves c_i tell us about the spread, respectively, nonlinearity, of the projection. Straight long 3D curves, for example, axes 6 and 8, show variables that are dominant (in terms of data variation) *and* well preserved (in terms of linearity) by the projection. Curved axes, for example, 3 and 1, show (local) nonlinearities of the projection. Short axes show dimensions along which data have less variation. The axes intersect in the projection centroid. Figure 2(b) shows the same dataset, projected with the FBDR force-based scheme in Tejada et al.[37] The extent and overall shape of the resulting 3D point cloud are very similar to the LAMP projection in Figure 2(a). Yet, the axes are now significantly more curved and entangled. This tells us directly that FBDR is less good than LAMP if we want to be able to "read" our nD variables along clearly separated directions in the projection space.

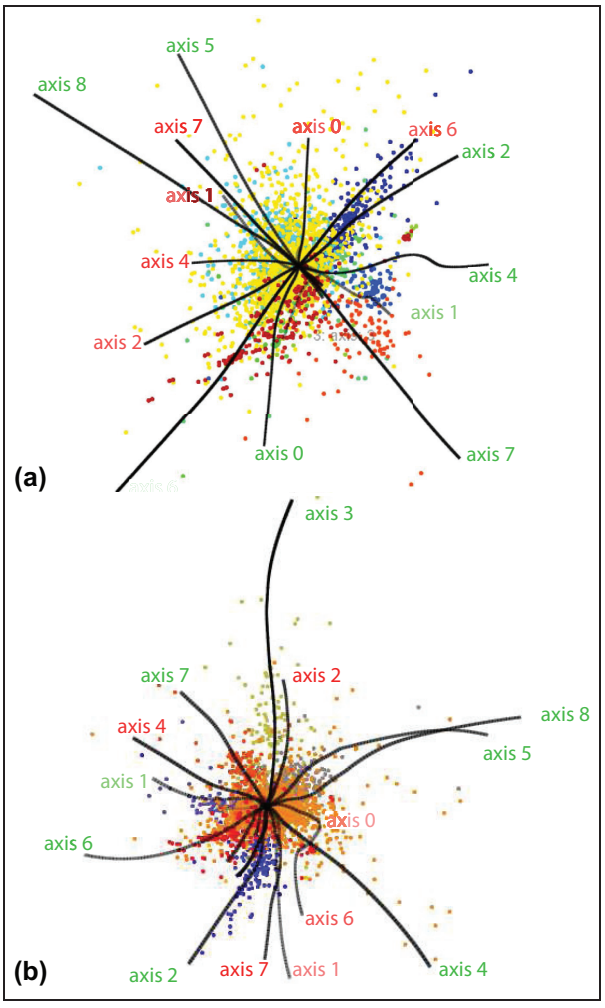

Figure 2. Adding curved biplot axes to the 3D projection in Figure 1: (a) LAMP projection and (b) FBDR projection.

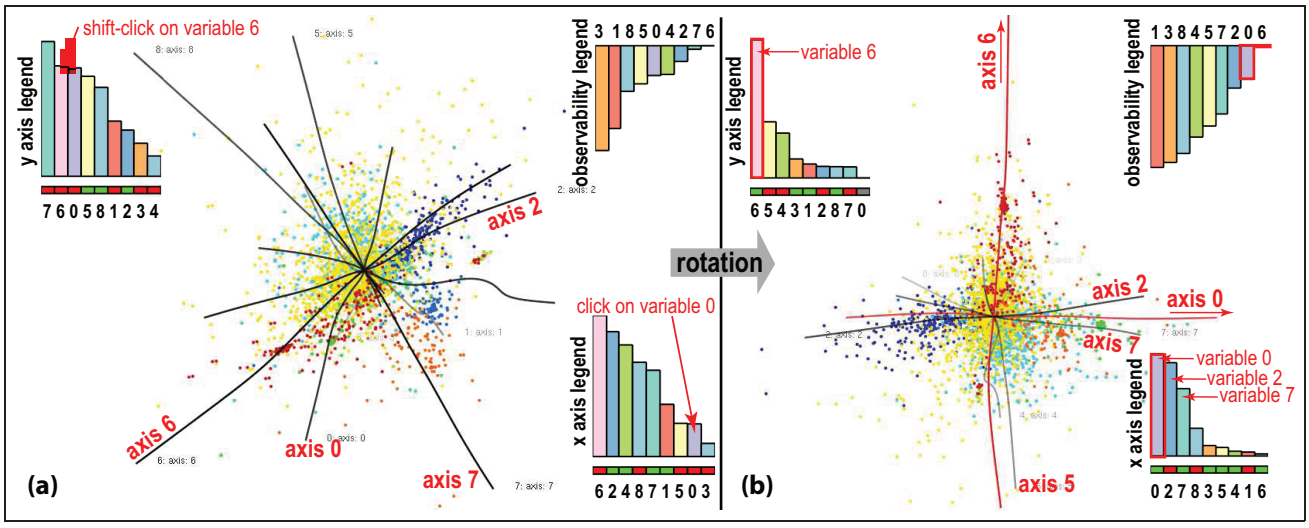

Figure 3. Axis legends. Two clicks in the left view will align variables 0 and 6 with the screen *x*- and *y*-axes, respectively, leading to the right view: (a) before alignment and (b) after alignment.

Enhanced axis legends

The users can view 3D projections from any viewpoint, using a virtual trackball to rotate, translate, and zoom the camera. For such a viewpoint, we denote the screen axes x and y by $\mathbf{x}_1$ and $\mathbf{x}_2$, and the view direction by $\mathbf{x}_3$. Given such a viewpoint, a key user question is "what can I see from here?" This question can be rephrased as follows: the variations in which original nD variables can we see best along screen axes $\mathbf{x}_1$ and $\mathbf{x}_2$? And which variables cannot that viewpoint show, because they get mapped along the view direction $\mathbf{x}_3$? Consider the analogy with the display of a simple 2D scatterplot of two variables, something that arguably most users are familiar with: The meaning of the screen x- and y-axes is clear—each such axis maps one of the two input variables. This is not so for our context, since (a) DR projects map many ($n \gg 2$) variables to three axes, so an axis will represent a "mix" of several variables and (b) we can freely choose any 3D viewpoint to look at the 3D DR projection. We propose to jointly address both (a) and (b), as follows.

Construction. To explain the screen axes, we use three bar charts, or axis legends (Figure 3), one for each of the axes $\mathbf{x}_j$, each having n bars for the n input variables. The height of the ith bar in the legend of $\mathbf{x}_j$ tells how much axis $\mathbf{x}_j$ shows the variation in the ith variable and is given by the absolute value of

$$h_i^j = \left((\mathbf{q}_S^i - \mathbf{q}_1^i) \cdot \mathbf{x}_j\right)\left(1 - \frac{\left|\|c_i\| - \left\|\mathbf{q}_S^i - \mathbf{q}_1^i\right\|\right|}{\|c_i\|}\right) \quad (4)$$

Here, $\|c_i\|$ is the length of the (curved) biplot axis c_i, computed as in section "Enhanced biplot axes." The first term in equation (4) is the projected length of c_i on screen axis $\mathbf{x}_j$. The high values hereof tell that we can easily see the spread of variable i along screen axis $\mathbf{x}_j$. The second term in equation (4) encodes the linearity of c_i. The high values hereof tell that the projection maps variable i to a straight line in the 3D projection space. The low values tell that variable i maps to a curved axis—so reading this variable along the *straight* screen axis $\mathbf{x}_j$ will be difficult. The high values of h_i^1 or h_i^2, that is, long bars in the $\mathbf{x}_1$ or $\mathbf{x}_2$ charts, are desirable, as they tell that the x or y screen axes can be used to directly read the variation in variable i. The high values of h_i^3 are undesirable, as they tell that variable i spreads mostly along the view direction, thus it is not observable from the current viewpoint. Hence, we call the $\mathbf{x}_3$ chart the *observability legend*. We also orient the bars of the x and y legends upward, and the bars of the observability legend downward, respectively (see Figure 3). This way, the upward-pointing direction of bars uniformly represents observability (of a variable) in all three legends.

The sign of h_i^j tells if variable i is mapped in the positive or negative direction of screen axis $\mathbf{x}_j$. We show this by a green ($h_i^j > 0$), red ($h_i^j < 0$), and gray ($h_i^j = 0$) box under each bar in the axis legends $\mathbf{x}_1$ and $\mathbf{x}_2$. This shows how a variable increases or decreases along a screen axis. For the view-direction axis $\mathbf{x}_3$, we do not show this sign, since data variations along this axis are, by definition, not visible from the current viewpoint. The bars in all three charts are colored to show the identity of the variables by a categorical

colormap created with ColorBrewer[65] and labeled by variable names (more about this next).

Sorting legends. We provide two modes to sort legend bars left to right. The first mode sorts bars alphabetically on their variable names, so bars for the same variable i appear at the same position in all three legends. This allows one to quickly visually scan and correlate the three legends to see how a given *variable* of interest is visible from the current viewpoint, that is, answer the question "Along which screen axis (x or y) can I best see this variable?" The second mode sorts bars in decreasing order of their $|h_i^j|$ values. This allows one to quickly see which are the best visible variables along a given screen *axis*, or answer the question "What does this screen axis show?" In this mode (see Figure 3), the bars for the same variable i may not appear at the same position in the three legends, but still have the same color and labels, to help correlation. This mode also addresses the case when we have a high-dimensional dataset, that is, n is large (tens or hundreds). Since legends are sorted, the most visible variables along the x and y screen axes are always the leftmost (and longest) bars of the x and y legends. If n exceeds a fixed preset $n_{max} = 20$, we only draw the first n_{max} longest bars. This ensures (a) that the $\mathbf{x}_1$ and $\mathbf{x}_2$ legends always show the n_{max} most visible variables from the current viewpoint, and (b) that bars are wide enough for their color and label annotations to be readable. For the $\mathbf{x}_3$ legend, the drawn bars tell us which are the *worst visible* variables from the current viewpoint. This helps answering the question "Which variables should I not try to analyze from the current viewpoint?" Summarizing, even when $n > n_{max}$, our three legends can tell us which are the best and worst visible n_{max} variables from any viewpoint. Apart from color coding, the bars in all three legends are linked, in both sorting modes, by brushing, similarly to the design proposed in Broeksema et al.[45] Whenever one moves the mouse pointer in a bar in a legend, this bar and the two other corresponding bars in the other two legends are highlighted. This way, one can quickly see how important a given variable of interest is along both x- and y-axes, and also how much of the variation in this variable cannot be observed from the current viewpoint, since it occurs along the view direction.

Linked views. We next use interactivity to support several exploration tasks. As the user changes the viewpoint, for example, by rotating the virtual trackball, axis legends dynamically change, so that one interactively sees how the viewpoint change affects what is mapped along the screen axes (see submitted video). Separately, we set the transparencies of the biplot axes c_i to the values $|h_i^3|$. The axes for variables with low $|h_i^3|$ values get emphasized (opaque), telling that their variables can be well read from the current viewpoint—see, for example, axes 7, 6, and 2 in Figure 3(a). Conversely, the axes for variables with high $|h_i^3|$ values are more transparent, telling that these variables are hard to read from the current viewpoint—see, for example, axis 5 in Figure 3(b).

Viewpoint selection. We further assist users to choose a good viewpoint by interactive-and-iterative axis alignment, as follows. Clicking any bar i in the $\mathbf{x}_1$ or $\mathbf{x}_2$ legends smoothly rotates the viewpoint to a new one where the biplot axis c_i for the clicked variable is best aligned, that is, has a maximal h_i^j value, with the clicked screen axis x_1 or x_2. Shift-clicking a second bar i' in the other legend (say, $\mathbf{x}_2$, if $\mathbf{x}_1$ was the first click) aligns variable i' with x_2, but constrains viewpoint rotation around x_1. This way, we get a viewpoint which best encodes the variation in two user-chosen variables—that is, creates the best possible scatterplot i versus i' allowed by the given DR projection—with only two clicks. Figure 3 illustrates this by showing how we align variables 0 and 6 with the screen x- and y-axes (Figure 3(a)). The resulting alignment (Figure 3(b)) also shows that axes 0 and 6 (marked red) are slightly curved, so that the projection is nonlinear. The y legend shows that the vertical data spread is mainly explained by variable 6. The x legend shows that the x spread is mainly explained by a mix of variables 0, 2, and 7, since the three longest bars in this legend have quite similar sizes. Since variable 0 is best aligned with the x-axis, by the alignment procedure, it means that variables 2 and 7 must *also* be well aligned with x too. It thus follows that variables 0, 2, and 7 project to (near) parallel axes in 3D, that is, they are strongly correlated. To check this, we brush the respective bars in the x plot, which highlights their biplot axes in 3D (apart from highlighting the corresponding bars in the three legends, as explained earlier). As shown in Figure 3(b), these are indeed correlated (the respective biplot axes are nearly parallel). Note that our x or y alignment tool is crucial for discovering correlations. Indeed, for the arbitrary viewpoint in Figure 3(a), the y bars for, for example, variables 7 and 6 are quite similar in length; yet, after alignment, we clearly see that variable 6 is orthogonal to variable 7.

Our approach is related to the legends in Broeksema et al.,[45] which show the variation in the nD variables along the screen x- and y-axes and the variation in the view direction (thus, not visible from a given viewpoint). Yet, important differences exist. First, the legends in Broeksema et al.[45] are static, as

their 2D projection is *predefined* by the SVD's two largest eigenvectors. Our dynamic legends help reading the nD variables from an interactively *user-chosen* viewpoint in 3D. For example, the x and y legends in Figure 3(a) show that viewpoint does not clearly let us read *individual* variables along the x and y screen axes, since many bars are long in these legends. After alignment, the legends significantly change (Figure 3(b)), telling us that x maps mainly a mix of variables 0, 2, and 7 and y maps mainly variable 6. We also see this in the observability legend (Figure 3(b), top right): the bars for variables 6, 0, 2, and 7 are shortest (in this order), telling that these variables are indeed almost fully captured by the xy screen space. In contrast, the bars for variables 1, 3, and 8 are longest; this indicates that these variables are poorly observable in the xy screen space for the current viewpoint, since they spread mainly in the z-direction. Second, while Broeksema et al.[45] orient bars in all three legends upward, we chose to orient the observability legend bars downward. This is in line with the fact that long bars in the observability legend are undesirable (they indicate variables we cannot see), while long bars in the x and y legends are desirable (they indicate variables we can see). Third, the computation of our bar heights is different. In Broeksema et al.,[45] these are the so-called loadings of the input n variables versus the two eigenvectors used for 2D projection. Computing loadings requires *explicit* knowledge of the DR method f used (SVD, in Broeksema et al.[45]). In contrast, we treat the DR method as a black box when creating our biplot axes (section "Enhanced biplot axes") and compute our bar heights separately as a function of the biplot axes' positions given by the current viewpoint (equation (4)). Hence, our biplot axes can be straight lines or curves, depending on the (non)linearity of f. In contrast, Broeksema et al.,[45] which uses the biplot setup in Abdi and Valentin,[44] assume a linear projection. Third, unlike Broeksema et al.,[45] sorting legends allows us to tell which variables can be best read along x and y, or worst read (because being orthogonal to the xy plane); discover variable correlations; and make legends scalable for large n values.

Viewpoint legend

Dynamic axis legends help seeing which variables are visible along the screen axes from a given viewpoint and also choose a good viewpoint to examine a *given* variable pair. Our next question is as follows: given a 3D DR projection, which relations (between *all* variable pairs) can we see well if we had time to go through *all* viewpoints?

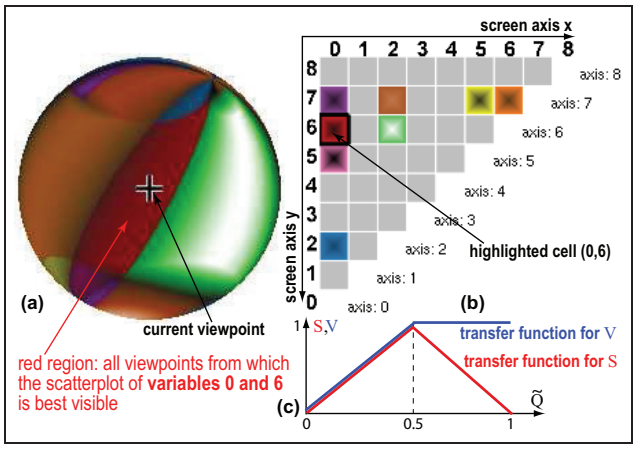

Figure 4. Legend for viewpoint shown in Figure 3 right.

We answer this question by a new interactive widget: the viewpoint legend (Figure 4). The widget uses a sphere S (Figure 4(a)); each point $\mathbf{v} \in S$ maps the viewpoint for the view direction $\mathbf{c} - \mathbf{v}$, where $\mathbf{c}$ is the center of S. Thus, S captures all possible viewpoints we can examine our 3D DR projection from. The central cross shows the current viewpoint. We uniformly sample S, in polar coordinates, by 400×400 viewpoints, and define the *quality* of each sample viewpoint $\mathbf{v}$ in terms of showing the variable pair $(i, j \neq i)$ as

$$q(\mathbf{v}, i, j) = \left\| (h_i^1, h_i^2) \times (h_j^1, h_j^2)^{\mathrm{T}} \right\| \tag{5}$$

Intuitively, q tells how well we can see from $\mathbf{v}$ the variation in variable i versus j, modulo all possible rotations of $\mathbf{x}_1$ and $\mathbf{x}_2$ in the view plane around the screen-normal axis $\mathbf{x}_3 = \mathbf{c} - \mathbf{v}$. This depends on how well the DR from $\mathbb{R}^n$ to $\mathbb{R}^3$, *and* the 3D-to-2D (screen) projection given by $\mathbf{v}$, captures this variation. Large $q(\mathbf{v}, i, j)$ values tell that the two biplot axes i and j are large *and* form a large angle (maximally, 90°) on the view plane. Such viewpoints are interesting to explore, as they show existing independent variable pairs which also have large spreads.

For each sample viewpoint $\mathbf{v}$, we compute the maximal value of q for all variable pairs (i, j)

$$Q(\mathbf{v}) = \max_{1 \leqslant i \leqslant n,\, 1 \leqslant j \neq i \leqslant n} q(\mathbf{v}, i, j) \tag{6}$$

For all $\mathbf{v}$, we also compute the normalized maximal quality $\tilde{Q}(\mathbf{v}) \in [0, 1] = Q(\mathbf{v}) / \max_{\mathbf{u} \in S} Q(\mathbf{u})$ and the variable pair $p(\mathbf{v}) = (i, j)$ which maximizes q at $\mathbf{v}$. Here, (i, j) are the variables that define the 2D scatterplot-like view we can best see from viewpoint $\mathbf{v}$. Next, we select the set P of $C = 8$ distinct variable pairs that have the largest values of q over all viewpoints $\mathbf{v} \in S$. P gives the C variable pairs we can best visualize from all

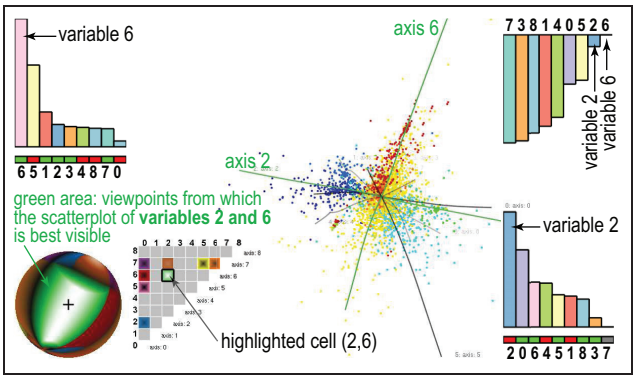

Figure 5. Selected viewpoint best showing scatterplot of variables 2 and 6.

possible viewpoints. We assign to each pair $p \in P$, thus to all C best-visible variable pairs, a distinct color $c(p)$, using a categorical colormap, and color the sphere points $\mathbf{v}$ as follows: If $p(\mathbf{v}) \in P$, we use for $\mathbf{v}$ the color $c(p)$, else we use the color gray. Next, we modulate the saturation S and brightness V of the assigned color at $\mathbf{v}$ by the quality $\bar{Q}(\mathbf{v})$ using the transfer functions shown in Figure 4(c). This effectively maps $\bar{Q}(\mathbf{v})$ to the shading of the sphere: low values are dark, mid-range values are saturated, and high values are white. Finally, we render this sphere using standard bilinear color interpolation over a quad mesh defined by our sample points $\mathbf{v}$.

To help interpreting the shaded sphere, we add a separate matrix-plot view (Figure 4(b)). Each variable pair (i,j) maps to a cell in this plot. The cells are colored using a two-color scheme, as follows. The first color is $c(p)$ for cells of pairs $p \in P$ and is gray for other cells. The second colormaps the value $\max_{\mathbf{v} \in S} q(\mathbf{v}, i, j)$ to a gray value between black and white. The cells are colored by linearly interpolating between the first color, assigned to the cell-border, and the second color, assigned to the cell center. The matrix plot thus shows both the C best-visible variable pairs, encoded by their respective colors, and the relative quality of different variable pairs, encoded by the brightness of their respective cell centers.

Figure 5 and the submitted video shows the added value of our viewpoint legend and matrix plot for our documents' dataset. We explore the viewpoint space interactively, as follows. Rotating the sphere changes the current viewpoint, which in turn dynamically updates the axis bar charts (section "Enhanced axis legends"). Conversely, rotating the 3D scatterplot (either manually or by axis-alignment animation, see section "Enhanced axis legends") turns the sphere in sync to show the newly selected viewpoint. The cell for the current viewpoint is highlighted on the matrix plot, so we can directly see which variable pair is best visible from that viewpoint, for example, (2, 6) in Figure 5. Clicking any cell (i, j) in the matrix plot smoothly rotates the viewpoint to one where the variable pair (i, j) is best visible, that is, goes to the viewpoint $\mathbf{v}$ where $q(\mathbf{v}, i, j)$ is maximal. This allows quickly navigating to such a viewpoint for any given variable pair—that is, constructs the best scatterplot $(i,j), \forall i \neq j$ by one click.

The viewpoint legend helps answering several questions, all related to choosing informative viewpoints for 3D DR projections, as follows:

Where from should I examine pair (i,j)*?* Large same-hue sphere zones, for example, the green one in Figure 5, show view-space areas from which the variable pair (i,j) is best visible. Looking up green in the matrix plot shows that this zone maps the variable pair $(2, 6)$.

Is there any good viewpoint for (i,j)*?* Small color zones show that some variable pairs are hard to see, since only few viewpoints allow that. This tells users not to expect to "create" such scatterplots from this DR projection, as this is very hard or even not possible. In other words, if understanding the correlation of such variable pairs is important, one should first change the DR projection.

How easy is to examine (i,j)*?* Large bright highlights in sphere zones show that the respective variable pair is easy to examine from many close viewpoints. Given our quality definition (equation (5)), this means that the spread of the values for these variables is large compared to other variables, *and* that the biplot axes' angles for these variables are large. This tells that creating scatterplots for the respective two variables is very easy—just move anywhere in the respective highlight and you will get the desired scatterplot. Moreover, the matrix-plot cell brightnesses tell us how easy is it to examine their respective variable pairs from *all* possible viewpoints: Bright cells tell that there is at least one viewpoint from where the respective pairs can be examined well (selectable by clicking that cell); dark cells tell that no such viewpoints exist.

What can I see from a given viewpoint? Highlights show viewpoints from where the variable pair given by the color around the highlight is best visible. Dark zones like the ones outside and close to the border of the green zone in Figure 5 tell that, from those viewpoints, there is no easy-to-see variable pair, since the *best* visible such pair has a low quality. Hence, such

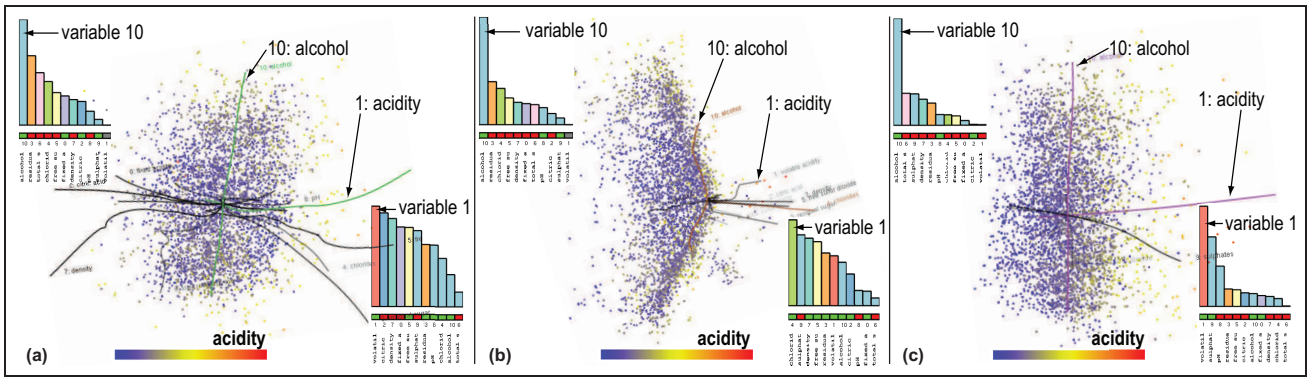

Figure 6. Selecting the best projection among three DR techniques using biplot axes and axis legends: (a) FBDR, (b) ISOMAP, and (c) LAMP. See section "Wine dataset: finding good DR projections."

zones tell that their respective viewpoints are arguably not useful for *any* visualization task.

How to relate more than two variables?. Color-zone *borders* show viewpoints where the best visible variable pair changes for small viewpoint rotations. These are typically bad viewpoints to examine a single variable pair. However, as we shall see in section "Multifield dataset: explaining projection shapes," these are good viewpoints to examine *groups* of three or more variables.

Applications

We next use our explanatory visualization techniques (enhanced biplot axes, axis legends, viewpoint legend) to explore 3D DR projections and aid in coarse correlations. They were constructed by three different DR methods, for four different datasets. By showing more datasets, we can easily explain how we address different kinds of questions with our tools, since each one has different data and, consequently, different questions related to it.

Wine dataset: finding good DR projections

This n =12 D dataset has 6497 points, each being a different sample of *vinho verde* wine.[66] The variables include chemical properties, for example, acidity, sugar and sulfur contents, chlorides, density, pH, and alcohol percentage. The last attribute is a user-assigned quality level. The tasks for this dataset involve finding correlations of the first 11 variables on one hand, and the quality on the other hand, over specific subsets of points; if found, such correlations could be next used to design automatic quality predictors.[66] To use DR for such tasks, we first must decide which DR method is best suited. One way for this is to select the DR method that minimizes aggregated projection errors, also called aggregated stress.[20] Yet, many state-of-the-art DR techniques will yield quite similar error values, so such aggregate errors are not discriminatory enough.

We consider here three DR methods: FBDR,[37] ISOMAP,[25] and LAMP[9] to project our dataset to 3D (other DR methods can be equally easily used). Figure 6 shows the obtained projections. For this dataset, these three projections yield very similar values for the normalized stress metric (equation (2)): 0.75 (ISOMAP), 0.81 (FBDR), and 0.83 (LAMP). Hence, how to say which DR method is best for discovering variable correlations? Showing our biplot axes helps us here (Figure 6). We see that FBDR and ISOMAP create, overall, quite twisted axes, unlike LAMP. Reading data values and/or finding if such axes are highly correlated (nearly parallel) or independent (nearly orthogonal) is clearly much easier if our axes are *straight* lines rather than curves. Our first finding is, thus, that LAMP is better for variable exploration in general.

However, the above does not imply that LAMP would be the best projection for more specific tasks, like exploring correlations of just two specific variables. Consider, for example, *alcohol* and *acidity*. We see that the *alcohol* axis is comparably straight for FBDR and LAMP—hence, we cannot yet rule out FBDR as a useful projection for this task. To study correlations against *alcohol*, we first click on the *alcohol* bar in the y legend to align it with the screen y-axis, in all three plots. Next, we use the same procedure to align *acidity* with the screen x-axis (one click on the *acidity* bar, x legend). For extra insight, we also color points by *acidity* values, using a blue–yellow–red divergent colormap. We now get several extra insights: First, we see that the x legend for FBDR has many bars of nearly equal size to *acidity*. Hence, either FBDR does not succeed in separating these variables

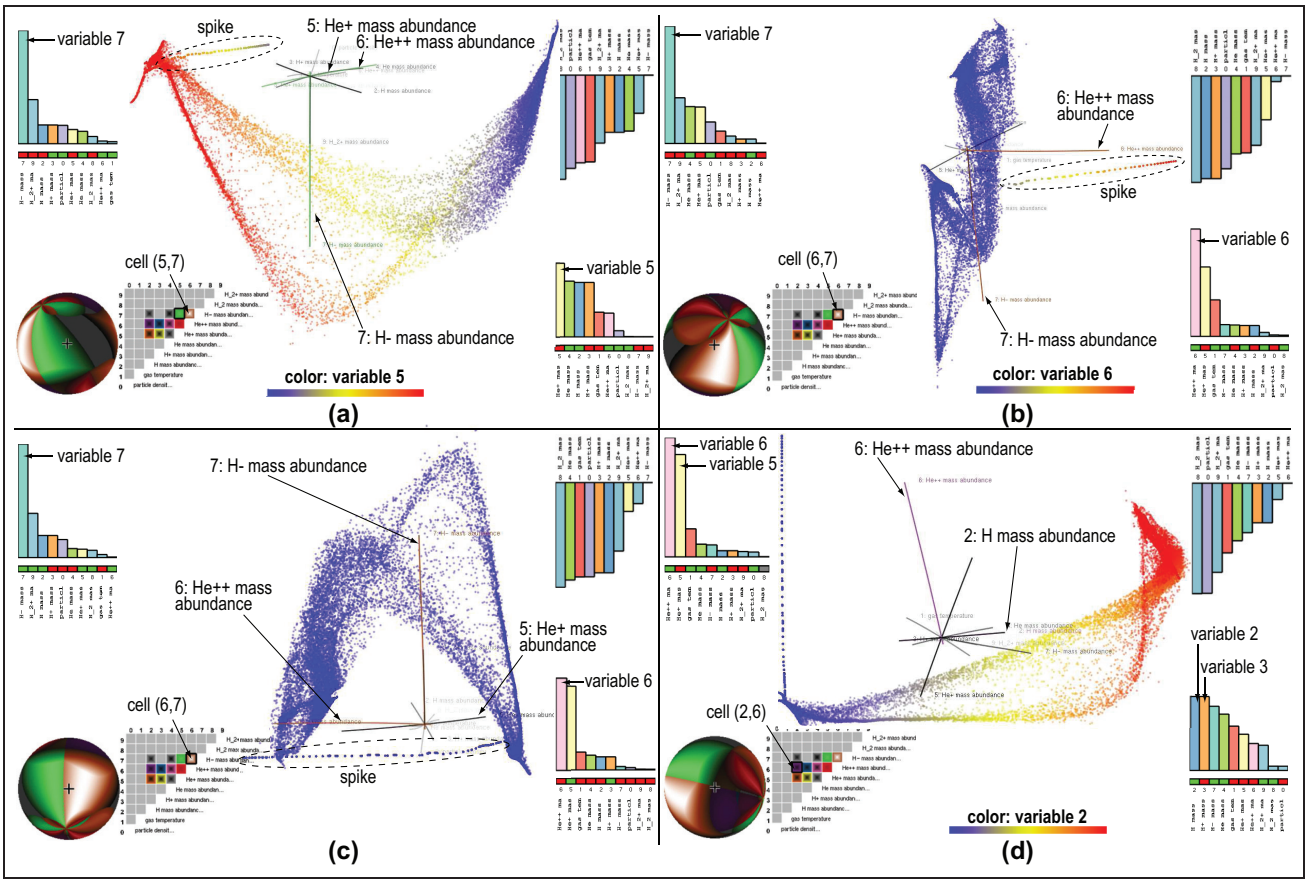

Figure 7. Explaining, in terms of variables, the shape of the 3D LAMP projection of 10-variate multifield simulation dataset (see section "Multifield dataset: explaining projection shapes"): (a) variables 5 and 7 aligned with screen axes *x* and *y*, (b) variable 6 aligned with screen axis *x*, (c) variables 6 and 7 aligned with screen axes *x* and *y*, and (d) variables 2 and 6 aligned with screen axes *x* and *y*.

during projection (which is bad) *or* we just discovered that these variables are highly correlated (which is a good finding). Yet, LAMP shows a clear exponential drop-off of the same bar lengths. Since LAMP's projection error is roughly equal to FBDR's, it means that the respective variables are *not* correlated; hence, the lack of separation in FBDR is a limitation of FBDR. Separately, we see that ISOMAP creates a twisted *acidity* axis and also shows a similar artificial correlation of variable projections along the *x* screen axis. Hence, we decide that LAMP is better than ISOMAP. Summarizing all above, we conclude that LAMP is the best of the three projections (LAMP, ISOMAP, and FBDR): it has a similar normalized stress metric, but succeeds best in creating straight, and well-separated, variable axes in 3D projection space.

Multifield dataset: explaining projection shapes

This $n = 10$ D dataset, from the IEEE Vis 2008 contest, encodes a time step of a multifield simulation dataset describing the formation of the early Universe.[67] The variables encode matter density, temperature, and concentrations of eight chemical species at 200,000 sample points. By freely rotating the 3D DR projection of this dataset (Figure 7), done using LAMP, we notice that the projection appears to be locally a 2D saddle-like manifold (point-cloud surface). We next want to better understand the shape of this surface and find the variables that determine it.

To do this, we turn on our biplot axes. We immediately notice that axis 7 is by far the longest—so variable 7 is important for explaining the projection's shape. Aligning variable 7 with the *y* screen axis shows that the projection appears to have a "saddle" shape (Figure 7(a)). We also see that axis 7 is nearly orthogonal to all other nine biplot axes. Hence, the *y* spread of the projection is mainly due to variable 7.

The viewpoint legend in Figure 7(a) shows next that variable 5 has a large variation which is largely independent on variable 7 (bright green zone on sphere; bright green cell in the matrix plot). To better explore the shape variation due to variables 5 and 7, we next

color points by variable 5, via the same colormap as in Figure 6. The result (Figure 7(a)) shows that the x stretch of our saddle shape is well explained by variable 5, which is high to the left and low to the right, as shown by both the colormap and the red cell under the variable 5 bar in the x legend. In this figure, we also notice an interesting "spike" line-like outlier in the top-left area. We can explain how this spike, as a specific internal substructure, aligns with specific axes by looking at them and see that the spike aligns best with axes 5 and 6. Iteratively aligning the x-axis (click on variable 5 bar in x legend, then click on variable 6 bar) shows that the spike best aligns with axis 6, as the x bar for variable 6 is largest. Figure 7(b) shows this viewpoint, with points colored by variable 6. We can now easily explain the spike as the locus of points having large variable 6 values (yellow … red). Indeed, all other points (on the saddle shape, not on the spike) have low variable 6 values (blue).

The viewpoint legend in Figure 7(b) shows that there are many viewpoints from which variables 6 and 7 project as independent axes (large brown area with bright highlight on sphere; bright highlight in the selected matrix-plot cell). Hence, variable 6 is indeed independent on variable 7, which was found the most important for explaining the saddle shape. Aligning variables 6 and 7 with the x- and y-axes, respectively (two clicks in the x and y legends), shows both the spike outlier and the saddle shape in a single view (Figure 7(c)). This view also shows that axes 5 and 6 are almost parallel, so variables 5 and 6 are highly correlated. We see this also in the viewpoint legend: the current viewpoint, which best shows variables 6 and 7, is very close to the brown–green zone border on the sphere. Also, both brown and green zones have very large bright highlights, *and* the brown–green border is also bright. Hence, most viewpoints that best show variables 6 and 7 *also* best show variables 5 and 7. We thus refine our earlier explanation of the saddle: this shape is best explained by variable 7 (in one direction) and variable 5 *or* 6 (in an orthogonal direction).

To explore variable 6 further, we look at its row in the matrix plot, and click the purple cell, to show its variation against variable 2. This aligns variables 2 and 6 with the x- and y-axes respectively, yielding the view in Figure 7(d). The x- and y-axis legends show now clearly that variables 5 and 6, respectively, 2 and 3, are highly correlated, since they have nearly equal *and* almost maximal bars.

As a final point, let us consider the effort required to explain the spike and saddle shapes present in the 3D scatterplot when using only classical projection–exploration tools such as the virtual trackball for rotation and the ability to color all projection points by the values of a chosen variable. Rotating the scatterplot so that we best see the spike outlier, that is, with the spike nicely aligned with the y-axis, takes about 2–3 min when using the virtual trackball. In contrast, this takes just two clicks on the x and y legends, as explained earlier. Finding that the spike is best explained by variable 6, while the saddle's spread in orthogonal direction to the spike is best explained by variable 2, requires, with standard tools, iteratively selecting each of the 10 variables to colormap the projection, detecting visually which is the strongest color gradient aligned with the spike, respectively, saddle, and memorizing this value. Using our tools, the color cycling is not required; we can directly see which variables align with specific scatterplot structures in terms of both biplot axes and axis legends.

Segmentation dataset: comparing 2D and 3D projections

Our third dataset has 2300 points with $n = 19$ variables. Each point describes a randomly chosen 3×3 pixel-block from seven manually segmented outdoor images, using 19 statistical image attributes, such as color mean, standard deviation, and horizontal/vertical contrast.[68] An extra manually set label attribute, not used in the DR projection, encodes the image type for each point.[9,14] The tasks for this dataset relate to designing automated image classifiers (using the 19 attributes) to match the manual classification (label attribute).[69]

Figure 8 shows this dataset using a 3D DR projection created by LAMP. By freely rotating this projection, with points colored by label values, we see that the longest biplot axis maps variable 0 (*region-centroid-col*). Aligning this axis with the y screen axis (click on *region-centroid-col* bar in the y legend) brings the viewpoint into a large red area on the viewpoint legend sphere. In the matrix plot, we see that red maps the variable pair $(0, 3)$. We next click this cell to go to the best viewpoint from which we can examine variables 0 and 3 (Figure 8(a)). The axis legends tell now that y explains almost only variable 0, while x explains mainly variable 3 (*short-line-density*). This viewpoint gives us two other interesting insights. First, we see that variable 0 has almost no correlation with the label-ID, that is, variable 0 takes virtually all values in its range for *any* single label-ID value. Next, by slightly rotating the viewpoint around the y-axis (variable 0), we see that axes 1–18 are located roughly in a plane orthogonal to axis 0. Together, the above tells us that variable 0 is not useful for classification, even though it is the most

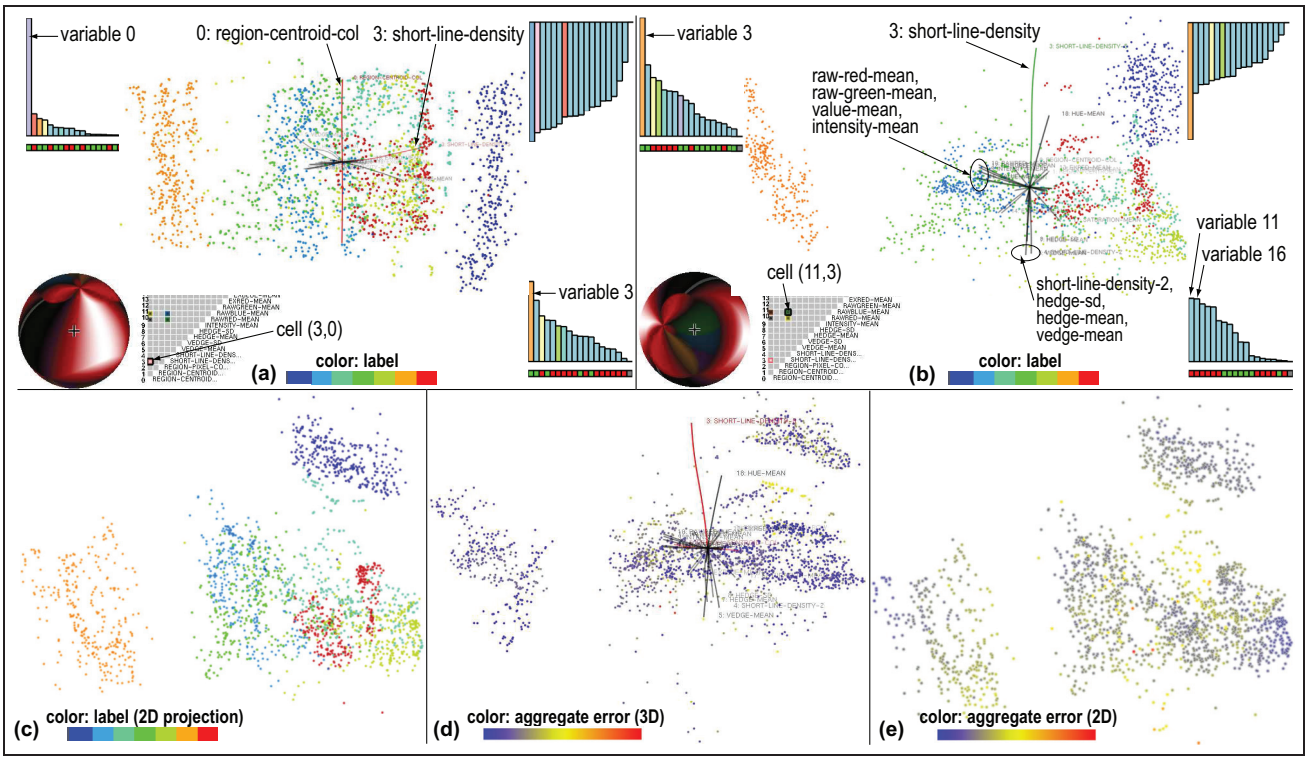

Figure 8. Visualization of 19-variate image segmentation dataset using (a, b, and d) 3D projections and (c and e) 2D projections. See section "Segmentation dataset: comparing 2D and 3D projections."

important in terms of variation and that the emerging clusters can be explained by variables 1–18.

To better understand the correlation of variables 1–18 with the label-ID, and thus get more insight into developing a classifier, we could next (a) remove variable 0 from the input dataset and redo the 3D DR projection (since we decided that this variable is not interesting), (b) view the current 3D projection from a suitable angle (to ignore the spread along axis 0), or (c) use a 2D DR projection rather than a 3D one (since Figure 8(a) suggests us that all interesting data variation occurs in a plane).

We examine next option (b). In the matrix plot in Figure 8(a), we see that all brightly colored cells are in columns 0 and 3, that is, the best viewpoints showing independent variable pairs always involve variables 0 and 3. The best such viewpoint (brightest red cell) maps variable pair $(0, 3)$ we just studied. We thus now choose to align biplot axis 3 with the y screen axis and biplot axis 0 (which we are not interested in) with the viewing direction z (Figure 8(b)). We now see a much clearer segregation of points by label-IDs into separate same-color clusters. This shows that there exist, indeed, correlations of the label-ID with attributes 1–18—thus, attributes 1–18 hold enough information to design a classifier. The biplot axes in Figure 8(b) help refining this insight. For instance, we see several strongly correlated variables: the group of axes pointing downward (*short-line-density-2*, *hedge-sd*, *hedge-mean*, and *vedge-mean*) all describe image edge features. The group of axes pointing to the left (*raw-red-mean*, *raw-green-mean*, *value-mean*, and *intensity-mean*) all capture means of the image colors. Correlating next these variables with the label variable (point colors) is a first step into explaining the clusters—for instance, we can now easily explain the isolated orange cluster as containing image blocks having highly saturated colors.

We next examine option (c). For this, we compute a 2D projection using again LAMP. Figure 8(c) shows the result, with points colored again by label-ID. The overall placement of clusters is quite similar, but not identical, to those in the 3D projection in Figure 8(b). To see which of these two images is a more faithful projection, we compute, for each point i, the aggregate normalized projection error $e_i^m \in [0, 1]$

$$e_i^m = \sum_{j \neq i} \left| \frac{d^m(\mathbf{q}_i, \mathbf{q}_j)}{\max_{i,j} d^m(\mathbf{q}_i, \mathbf{q}_j)} - \frac{d^n(\mathbf{p}_i, \mathbf{p}_j)}{\max_{i,j} d^n(\mathbf{p}_i, \mathbf{p}_j)} \right| \quad (7)$$

Here, d^n, d^m, $\mathbf{p}$, and $\mathbf{q}$ have the same meaning as in equation (2). The error e_i^m, $m \in \{2, 3\}$, tells how well the mD projection of a point i approximates its placement in $\mathbb{R}^n$ from the perspective of its distances to all

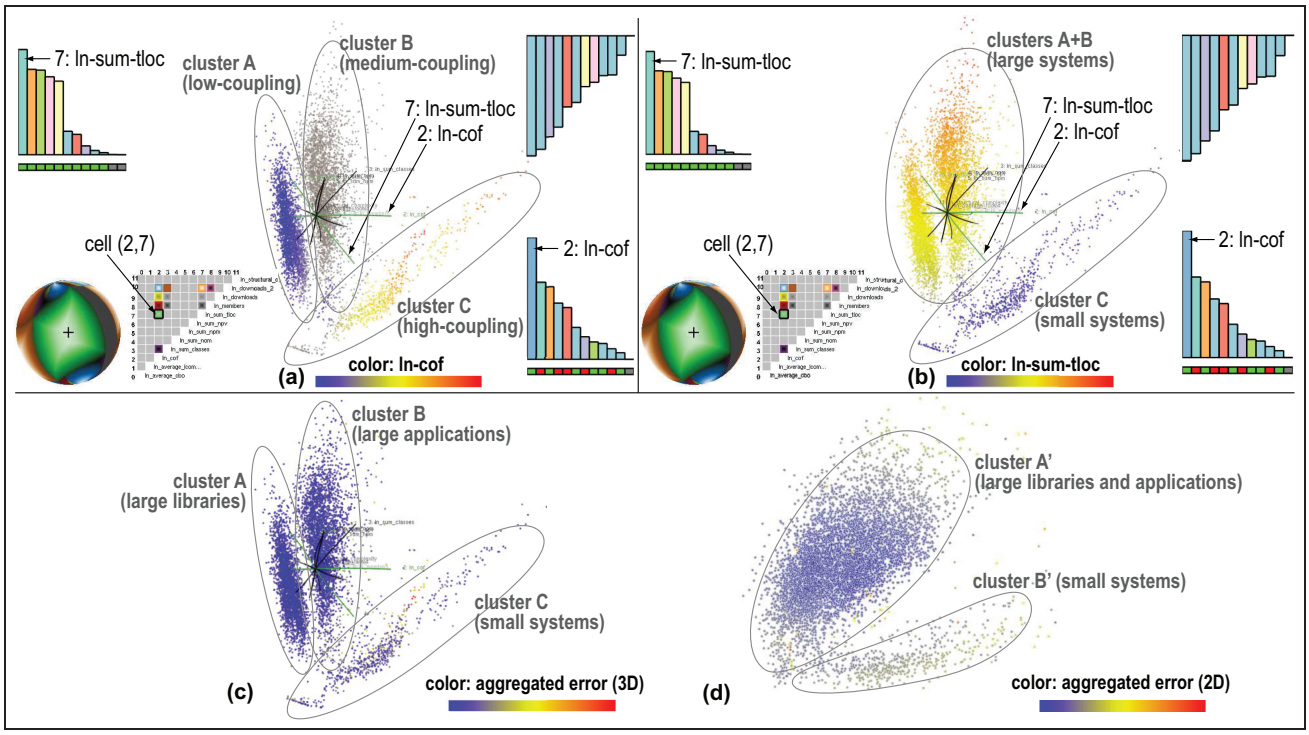

Figure 9. Visualization of 12-variate software metrics dataset using (a–c) 3D DR projections and (d) corresponding 2D DR projection. See section "Software dataset: finding meaningful clusters."

other points $j \neq i$. More details on this metric are given in Martins et al.[20]

Figure 8(d) and (e) shows the errors e_i^3 and e_i^2 for the 3D and 2D projections in Figure 8(b) and (c), respectively, colormapped as in Figure 7. While both projections "spread" errors quite uniformly over all points, and do not create any extreme errors, we see that e_i^3 is overall lower than e_i^2. So, the 3D LAMP projection preserves the original nD distances better than 2D LAMP. Hence, for this dataset, using a 3D DR projection, with a suitably chosen viewpoint provided by our exploratory tools (Figure 8(b)), is better than using a 2D DR projection generated by the same DR technique. This is not entirely surprising, once we understand Figure 8(a): the 2D projection has to accommodate the large variation in variable 0 in the same (limited) 2D space used to project all other 18 variables. In contrast, the 3D projection can freely spread all this variation along a separate spatial dimension. Thus, examining the 3D projection from the *single* viewpoint shown in Figure 8(b)—which is roughly equivalent to a 2D projection of variables 1–18—is better, error-wise, than using a 2D projection of the entire dataset.

Note that the use of our explanatory tools is very different in this use-case than in the one discussed in section ""Multifield dataset: explaining projection shapes." Indeed, in section ""Multifield dataset: explaining projection shapes," we used our tools to select a *variety* of viewpoints, which next helped us explain the projection's shape in terms of variables. In the example here, we used our tools to decide that we can best explore the projection from a *single* viewpoint, and next to choose this viewpoint.

Software dataset: finding meaningful clusters

Our fourth and final example uses a set of 6733 open-source software projects written in C. The source code of each project was downloaded to compute 11 code quality metrics as averages over the project's code files. A 12th metric gives the number of downloads of each project.[70] This yields a $n = 12$D dataset with 6733 points. While Meirelles et al.[70] explored the statistical correlation of project quality with download count, we want to get finer-grained insights of the types of projects involved in the studied code-base collection.

For this, we use a 3D LAMP projection of our 12D dataset (Figure 9(a)–(c)). We first find the best visible variable pair from any 3D viewpoint, by clicking the bright green cell in the matrix plot in Figure 9(a). This gives us variables 2 (*ln-cof*, or average coupling-factor, that is, the number of function calls between files[71]) and 7 (*ln-sum-tloc*, or total number of lines-of-code).

Next, we align axis 2, the longest of these two biplot axes, with screen x-axis (Figure 9(a)). We notice two well-separated point clusters (A and B), which spread orthogonally to biplot axis 2 (*ln-cof*). To understand what these mean, we color points by variable 2. This shows that clusters A and B contain points having two different ranges of *ln-cof* values: A contains low-coupling systems (such as libraries), while B contains medium-coupling systems (such as full applications). We also see here a third cluster (C) formed by very high *ln-cof* points. These points are also orthogonal to axis 7 (*ln-sum-tloc*). Hence, to check whether variable 7 explains cluster C, we next color points by variable 7 (Figure 9(c)): we now indeed see that nearly all points in C have low values of variable 7, and all points in A and B have high values for variable 7. Thus, cluster C contains highly coupled, small-scale software systems (small applications). Summarizing, we found that our 3D DR projection groups our 6733 software projects in three classes: large software projects (high values for *ln-sum-tloc*), further split by project type into libraries (A), and full applications (B), and C, containing small applications (low values for *ln-sum-tloc*). The entire 3D analysis requires just three clicks: one to align the screen x- and y-axes with the best separated variables *ln-cof* and *ln-sum-loc* and two further clicks to color points by values of these variables, respectively.

As for the segmentation dataset (section "Segmentation dataset: comparing 2D and 3D projections"), we want next to see whether a 2D DR projection could give us the same insight given by our 3D DR projection, that is, that our 6733 software projects can be grouped into three distinct classes. For this, we first color our 3D projection points by their aggregated projection error e_i^3 (equation (7)). Figure 9(c) shows this. Next, we do an nD-to-2D projection (also by LAMP) and color it by its projection error e_i^2 (Figure 9(d)). Comparing Figure 9(c) and (d), we see that, like for our segmentation dataset, both e_i^3 and e_i^2 are uniformly spread over their respective projections, with $e_i^3 < e_i^2$ on average. However, in contrast to the segmentation dataset, we see that the 3D DR projection creates three clusters (explained by variables *ln-cof* and *ln-sum-tloc*, as discussed); the 2D projection creates only two clusters A' and B' (Figure 9(d)). By manual brushing of the displayed data points, we found that A' contains a mix of points in A and B (large libraries *and* applications), while B' roughly corresponds to C (small systems). This is also visible in Figure 9(a) and (b): rotating the viewpoint along the view sphere, and looking at the variation in the axes legends (or alternatively, at the biplot axes), we find no viewpoint in which $n - 1$ axes reside in, or close to, a plane. Thus, three projection dimensions are truly needed to show the data variation that encodes the three clusters—that is, we need a 3D DR projection to obtain a view that segregates our software systems into three clusters corresponding to large libraries, large applications, and small systems. A 2D DR projection can only segregate software projects into large and small systems, but not segregate based on the coupling type (applications vs libraries).

Discussion

Several points are relevant to discuss, as follows.

Scope

The effectiveness of our techniques depends, of course, on the quality of the DR projection and nature of the underlying nD dataset. If the projection captures distinct, well-separated, patterns in mD, our techniques will help explain the relationships of these patterns with the original n dimensions, and next choose good viewpoints to examine them. If the DR projection is suboptimally done, or if the input dataset does not exhibit any clearly segregated patterns, our techniques provide little additional insight in the data. So, our scope is to help users explain patterns, through in course correlations, the projected data in terms of the original variables, *if* such patterns exist in mD. If patterns are absent, one should use complementary techniques, outside the scope of our work, to improve the DR projection being used, for example, Martins et al.[20] Separately, if the nD data are clearly segregated into clusters *and* if one only wants to find such clusters, rather than the more fine-grained task of explaining spreads in the data or correlations of specific variables, then state-of-the-art clustering methods are the optimal tool.

Our key added value is for 3D DR projections, where viewpoint and navigation choices critically affect the obtained insights.[6,17] Let us explain this. Our final 2D view can be seen as being created by "concatenating" an nD-to-3D DR projection (P_{n3}) with a 3D-to-2D screen projection (P_{32}). As discussed, P_{n3} typically has a lower error than a direct nD-to-2D DR projection (P_{n2}). Our tools allow understanding and controlling the error given by P_{32}; in contrast, a P_{n2} does not allow any kind of similar error control. For instance, we can interactively change P_{32} to select which variables, or dataset parts, are finally best visible. If, in any view, one axis is small, it means that this variable is not visible in that view, so we cannot reason about it. However, we can rotate the view by aligning this axis to the 2D screen and next interpret the resulting view, thereby obtaining the best view that shows the spread of that variable. In particular, a P_{32} using the two longest axes is as precise as a direct P_{n2}, in terms of stress

error. Note that this cannot be done with a direct P_{n2}: if an axis is small in such a projection, we cannot do *anything* about that, and no interpretation of data variations along that axis is possible. Regarding occlusion, our solution is as good as, or better than, using a direct P_{n2}: in our 2D views, occlusion means that 3D points overlap along view lines; yet, we can choose other 3D viewpoints where such overlaps are decreased; in contrast, such overlaps occur in any P_{n2} too, and we cannot do anything to decrease them in that case.

Generality

Our techniques work directly with any (non)linear DR technique that projects n variables to $m = 3$ dimensions, without needing to modify, or access the internals of, the DR technique. This is unlike Oeltze et al.,[13] Greenacre,[42] Abdi and Valentin,[44] and Broeksema et al.[45] which need to know that the DR being used is principal component analysis (PCA) or SVD to compute loading values. Our examples shown here use LAMP, ISOMAP, and FBDR as DR techniques. We have equally easily used LSP[6] and PLMP.[14] Other DR projection techniques can be equally easily used, with no changes to our proposal.

Scalability

Our methods are simple to implement and computationally scalable: we only need to apply the chosen DR projection to a small set of sample points distributed along the input variables (section "Enhanced biplot axes"). For a dataset of D variables, N data points, and a number of n_{max} variables shown in the proposed legends, the complexities are $O(D)$ for the biplot calculation, $O(n_{max})$ for the axis legends, and $O(n_{max}^2)$ for the viewpoint legend, respectively. The memory complexity of the entire set of techniques is $O(N \cdot D)$, that is, equal to the size of the dataset to be stored. Visually, our axes legends, biplot axes, and viewpoint legend scale well up to roughly $n_{max} = 20$ variables, in line with other multivariate visualization techniques.[1,13,45,61] When the input dataset has more variables, axis legends automatically show the n_{max} most visible variables for the current viewpoint, which is the best we can do in such situations (section "Enhanced axis legends").

Comparison

Our axis alignment and viewpoint legends have some similarities (and differences) with RTD.[1] Our axis alignment (section "Enhanced biplot axes") and best viewpoint tools (section "Viewpoint legend") resemble the scatterplot-matrix cells in the sense of selecting "interesting" variable pairs. Yet, while RTD defines these configurations as variable pairs mapped to Cartesian scatterplots, we define these as viewpoints in a 3D space *given* by the DR projection that can best highlight variable combinations of interest. Since we cannot control the DR projection, our viewpoints can show orthogonal biplot axes, and also slanted and/or curved axes of different lengths. Also, our viewpoints show, by construction, *all* projected axes, rather than a fixed subset of two. Finding a good data-exploration sequence is equivalent, in our case, to find a navigation path between highlights on the viewpoint legend sphere. The main added value of the viewpoint legend is that it shows *all* possible viewpoints in-between these highlights.

Technical details

Our categorical, continuous colormaps, and transfer function choices (section "Viewpoint legend") are, of course, open. For instance, one can customize the categorical colormap used in the axis legends (section "Enhanced axis legends") to mark specific variables of interest, which one needs to pay particular attention during the analysis, with salient colors or colors having an application-specific semantics. Alternatively, one could select an axis legend, colormap its bars using a sequential or ordinal colormap, and next compare this legend color-wise with the other two axis legends to reason about variable correlation or orthogonality. Yet other alternatives may exist for specific user groups and work domains. We used simple and well-known presets for these designs precisely to make it easier to separate our contributions from such specific design elements.

Evaluation

We evaluated the proposed techniques on nine datasets (300–200,000 points, and 6–25 variables). Learning to interpret the axes biplots, axes legends, and viewpoint legend was perceived to be pleasant and easy, mainly due to the fact that all these visualizations are interactive and dynamically change as the user rotates the viewpoint. Besides the selection of the variable used to color points, our techniques do not require any explicit parameter user setting. Compared to classical 2D scatterplots, our techniques need additional time to learn them (around 20 min, as observed by explaining them to nine users not involved in this work)—which is in line with learning times reported in Elmqvist et al.[1] and Broeksema et al.[45] for similar tasks and user counts. Users found the biplot axes easiest to understand and use, arguably due to the fact that similar axes appear in many types of plots. The interactive axis

alignment described in section "Enhanced axis legends" was also found simple to understand and use, as it requires basically two clicks in the desired bars of the *x*- and *y*-axis legends. Using the viewpoint legend was perceived as the most complicated, as this widget requires memorizing the appearance of several large same-color areas on the surface of the sphere while interactively rotating the viewpoint. We acknowledge that these findings need more refinement and validation, for example, in terms of a controlled user study.

Limitations

Large 3D DR scatterplots inherently generate occlusion which, even with transparency and interaction, can be hard to disambiguate. Biplot axes for a few highly nonlinear projections (e.g. force-based methods[31,37]) are highly curved. Yet, such methods are not preferred, precisely because of their error rates and the difficulty of finding globally good viewpoints, and thus affect our overall proposal only marginally. Our tools do not aim to fully remove interactive trial-and-error exploration, such as brushing and viewpoint selection. Their added value is to make interaction more *targeted* toward a given goal—for example, when (slightly) changing a viewpoint, one immediately sees the effect on the axis biplots, axis legends, and viewpoint legend, and thus can better estimate what to expect to see when turning the viewpoint this or that way; when one wants to examine one or two specific variables in context, we allow doing this by just two clicks on the axis-legend bars for those variables. Separately, we note that our examples in section "Applications" do not imply that 3D projections are always bet for addressing all related tasks: rather, we show how 3D DR projections, *if* chosen for the sake of minimizing distance errors, can be made more effective as compared to raw 3D scatterplots.

Conclusion

We have presented a set of interactive visualizations that help users explore and explain 3D DR projections of high-dimensional data. Our methods, realized as linked views, explain the meaning of projected dimensions in terms of original variables; show projection nonlinearities and correlations (or lack thereof) for these variables; help finding good viewpoints from which given variable pairs can be best explored; and quickly show which variable pairs can be explored from any possible viewpoint. Globally, our techniques aim to help users interpret raw 3D projections in typical *xy* scatterplot terms. Our techniques are easy to implement, scale well computationally and visually, and can be added in a non-intrusive way to any DR technique as extra aids to classical brushing and color-mapping explanatory tools.

Future work targets enhancing the insight given by our explanatory visualizations, by studying how the local nonlinearity of projections, and local projection errors, can be better and more intuitively conveyed for large 3D projections. Validating the value of our visualizations via user studies is a second important future work topic.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Elmqvist N, Dragicevic P and Fekete JD. Rolling the dice: multidimensional visual exploration using scatterplot matrix navigation. *IEEE T Vis Comput Gr* 2008; 14: 1141–1148.
2. Heinrich J and Weiskopf D. Continuous parallel coordinates. *IEEE T Vis Comput Gr* 2009; 15(6): 1531–1538.
3. Cui W, Wu Y, Liu S, et al. Context-preserving, dynamic word cloud visualization. *IEEE Comput Graph* 2010; 30: 42–53.
4. Endert A, Flaux P and North C. Semantic interaction for visual text analytics. In: *Proceedings of the ACM (CHI 12)*, Austin, TX, 5–10 May 2012, pp. 324–333. New York: ACM.
5. Olsen K, Korfhage R, Sochats K, et al. Visualization of a document collection: the VIBE system. *Inform Process Manag* 1993; 29(1): 69–81.
6. Paulovich FV, Nonato L, Minghim R, et al. Visual mapping of text collections through a fast high precision projection technique. In: *Proceedings of the IEEE information visualization*, London, 5–7 July 2006, pp. 282–290. New York: IEEE.
7. Yi J, Melton R, Stasko J, et al. Dust & Magnet: multivariate information visualization using a magnet metaphor. *Inform Visual* 2005; 4(4): 239–256.
8. Faloutsos C and Lin K. FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In: *Proceedings of the ACM SIGMOD international conference on management of data*, San Jose, CA, 22–25 May 1995, pp. 163–174. New York: ACM.
9. Joia P, Paulovich FV, Coimbra D, et al. Local affine multidimensional projection. *IEEE T Vis Comput Gr* 2011; 17: 2563–2571.
10. Chen Y, Wang L, Dong M, et al. Exemplar-based visualization of large document corpus. *IEEE T Vis Comput Gr* 2009; 15: 1161–1168.
11. Paulovich FV, Nonato LG, Minghim R, et al. Least square projection: a fast high-precision multidimensional projection technique. *IEEE T Vis Comput Gr* 2008; 14(3): 564–575.
12. Daniels J, Anderson E, Nonato L, et al. Interactive vector field feature identification. *IEEE T Vis Comput Gr* 2010; 16: 1560–1568.

13. Oeltze S, Doleisch H, Hauser H, et al. Interactive visual analysis of perfusion data. *IEEE T Vis Comput Gr* 2007; 13(6): 1392–1399.
14. Paulovich FV, Silva C and Nonato L. Two-phase mapping for projecting massive data sets. *IEEE T Vis Comput Gr* 2010; 16: 1281–1290.
15. Poco J, Eler DM, Paulovich FV, et al. Employing 2D projections for fast visual exploration of large fiber tracking data. *Comput Graph Forum* 2012; 31: 1075–1084.
16. Newby G. Empirical study of a 3D visualization for information retrieval tasks. *J Intell Inf Syst* 2002; 18(1): 31–53.
17. Sedlmair M, Munzer T and Tory M. Empirical guidance on scatterplot and dimension reduction technique choices. *IEEE T Vis Comput Gr* 2013; 19(12): 2634–2643.
18. Westerman S, Collins J and Cribbin T. Browsing a document collection represented in two- and three-dimensional virtual information space. *Int J Hum Comput St* 2005; 62(6): 713–736.
19. Jolliffe IT. *Principal component analysis*. New York: Springer, 2002.
20. Martins R, Coimbra D, Minghim R, et al. Visual analysis of dimensionality reduction quality for parameterized projections. *Comput Graph* 2014; 41: 26–42.
21. Poco J, Etemadpour R, Paulovich FV, et al. A framework for exploring multidimensional data with 3D projections. *Comput Graph Forum* 2011; 30(3): 1111–1120.
22. Torgeson WS. Multidimensional scaling of similarity. *Psychometrika* 1965; 30: 379–393.
23. Roweis ST and Saul LK. Nonlinear dimensionality reduction by locally linear embedding. *Science* 2000; 290(5500): 2323–2326.
24. Silva V and Tenenbaum J. Global versus local methods in nonlinear dimensionality reduction. In: Becker S, Thrun S and Obermayer K (eds) *Advances in neural information processing systems*, vol. 15. Cambridge, MA: MIT Press, 2003, pp. 705–712.
25. Tenenbaum J, de Silva V and Langford J. A global geometric framework for nonlinear dimensionality reduction. *Science* 2000; 290(5500): 2319–2323.
26. Silva V and Tenenbaum J. *Sparse multidimensional scaling using landmark points*. Stanford, CA: Stanford University, 2004.
27. Brandes U and Pich C. Eigensolver methods for progressive multidimensional scaling of large data. In: *Proceedings of the graph drawing*, Karlsruhe, 18–20 September 2007, pp. 42–53. Berlin: Springer.
28. Gansner ER, Koren Y and North S. Graph drawing by stress majorization. In: *Proceedings of the 12th international symposium GD*, New York, 29 September–2 October 2004, pp. 239–250. Berlin: Springer.
29. Sammon JW. A nonlinear mapping for data structure analysis. *IEEE T Comput* 1969; C-18: 401–409.
30. Bronstein M, Bronstein A, Kimmel R, et al. Multigrid multidimensional scaling. *Numer Linear Algebr* 2006; 13: 149–171.
31. Pekalska E, de Ridder D, Duin R, et al. A new method of generalizing Sammon mapping with application to algorithm speed-up. In: *Proceedings of the ASCI*, 1999, pp. 221–228.
32. Eades P. A heuristic for graph drawing. *Congr Numer* 1984; 42: 149–160.
33. Chalmers M. A linear iteration time layout algorithm for visualising high dimensional data. In: *Proceedings of the IEEE visualization*, San Francisco, CA, 27 October–1 November 1996, pp. 127–131. New York: IEEE.
34. Frishman Y and Tal A. Multi-level graph layout on the GPU. *IEEE T Vis Comput Gr* 2007; 13: 1310–1319.
35. Ingram S, Munzner T and Olano M. Glimmer: multilevel MDS on the GPU. *IEEE T Vis Comput Gr* 2009; 15(2): 249–261.
36. Jourdan F and Melangon G. Multiscale hybrid MDS. In: *Proceedings of the IEEE information visualization*, London, 14–16 July 2004, pp. 388–393. New York: IEEE.
37. Tejada E, Minghim R and Nonato LG. On improved projection techniques to support visual exploration of multidimensional datasets. *Inform Visual* 2003; 2(4): 218–231.
38. Fadel S, Fatore F, Duarte F, et al. LoCH: a neighborhood-based multidimensional projection technique for high-dimensional sparse spaces. *Neurocomputing* 2015; 150: 546–556.
39. Sorzano C, Vargas J and Pascual-Montano A. A survey of dimensionality reduction techniques, 2014, http://arxiv.org/pdf/1403.2877
40. Van der Maaten L, Postma E and van den Herik H. Dimensionality reduction: a comparative review. *J Mach Learn Res* 2009; 10(1): 66–71 (Extended version), http://www.iai.uni-bonn.de/~jz/dimensionality-reductions_comparativeseview.pdf
41. Gower JC and Hand DJ. *Biplots,* vol. 54. Boca Raton, FL: CRC Press, 1995.
42. Greenacre M. *Biplots in practice*. Bilbao: Fundacion BBVA, 2010.
43. Gower J, Lubbe S and Roux N. *Understanding biplots*. Chichester: Wiley, 2011.
44. Abdi H and Valentin D. Multiple correspondence analysis. In: Salkind N (ed.) *Encyclopedia of measurement and statistics*. Thousand Oaks, CA: SAGE, 2007, pp. 652–658.
45. Broeksema B, Telea A and Baudel T. Visual analysis of multidimensional categorical data sets. *Comput Graph Forum* 2013; 32(8): 158–169.
46. Geng X, Zhan D and Zhou Z. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE T Syst Man Cy B* 2005; 35(6): 1098–1107.
47. Anand A, Wilkinson L and Dang T. Visual pattern discovery using random projections. In: *Proceedings of the IEEE visual analytics science and technology (VAST)*, Seattle, WA, 14–19 October 2012, pp. 43–52. New York: IEEE.
48. Lewis J, van der Maaten L and de Sa V. A behavioral investigation of dimensionality reduction. In: *Proceedings of the cognitive science society*, 2012, pp. 671–676.
49. Schreck T, von Landesberger T and Bremm S. Techniques for precision-based visual analysis of projected data. *Inform Visual* 2010; 9(3): 181–193.

50. Aupetit M. Visualizing distortions and recovering topology in continuous projection techniques. *Neurocomputing* 2007; 10(7–9): 1304–1330.
51. Bertini E, Tatu A and Keim D. Quality metrics in high-dimensional data visualization: an overview and systematization. *IEEE T Vis Comput Gr* 2011; 17(12): 2203–2212.
52. Piringer H, Kosara R and Hauser H. Interactive F + C visualization with linked 2D/3D scatterplots. In: *Proceedings of the second international conference on coordinated and multiple views in exploratory visualization (CMV IEEE)*, London, 13 July 2004, pp. 49–60. New York: IEEE.
53. Sanftmann H and Weiskopf D. Illuminated 3D scatterplots. *Comput Graph Forum* 2009; 28(3): 642–651.
54. Hurter C, Taylor A, Carpendale S, et al. Color tunneling: interactive exploration and selection in volumetric datasets. In: *Proceedings of the IEEE Pacific visualization symposium*, Yokohama, Japan, 4–7 March 2014, pp. 327–335. New York: IEEE.
55. Claessen J and Van Wijk JJ. Flexible linked axes for multivariate data visualization. *IEEE T Vis Comput Gr* 2011; 17(12): 2310–2316.
56. Tavanti M and Lind M. 2D vs 3D, implications on spatial memory. In: *Proceedings of the IEEE information visualization*, San Diego, CA, 22–23 October 2001, pp. 139–145. New York: IEEE.
57. Westerman S and Cribbin T. Mapping semantic information in virtual space: dimensions, variance and individual differences. *Int J Hum Comput St* 2000; 53(5): 765–787.
58. Fabricant S. *Spatial metaphors for browsing large data archives*. PhD Thesis, University of Colorado Boulder, Boulder, CO, 2000.
59. Dang T, Wilkinson L and Anand A. Stacking graphic elements to avoid over-plotting. *IEEE T Vis Comput Gr* 2010; 16(6): 1044–1052.
60. Sanftmann H. 3D visualization of multivariate data, 2014, http://elib.uni-stuttgart.de/opus/volltexte/2012/7807
61. Chan Y, Correa C and Ma K. Regression cube: a technique for multidimensional visual exploration and interactive pattern finding. *ACM Trans Interact Intell Syst* 2014; 4(1): 7.
62. Sanftmann H and Weiskopf D. 3D scatterplot navigation. *IEEE T Vis Comput Gr* 2012; 18(11): 1969–1978.
63. Paiva J, Schwartz W, Pedrini H, et al. Semi-supervised dimensionality reduction based on partial least squares for visual analysis of high dimensional data. *Comput Graph Forum* 2012; 31(3): 1345–1354.
64. Salton G. *Introduction to modern information retrieval*. New York: McGraw-Hill, 1986.
65. Brewer C and Harrower M. ColorBrewer, 2014, http://www.colorbrewer.org
66. Cortez P, Cerdeira A, Almeida F, et al. Modeling wine preferences by data mining from physicochemical properties. *Decis Support Syst* 2009; 47(4): 547–553.
67. Norman M and Whalen D. IEEE visualization 2008 contest data, 2013, http://sciviscontest.ieeevis.org/2008
68. Frank A and Asuncion A. UCI machine learning archive, 2013, http://www.ics.uci.edu/~mlearn
69. Paulovich FV, Eler D, Poco J, et al. Piecewise laplacian-based projection for interactive data exploration and organization. *Comput Graph Forum* 2011; 30(3): 1091–1100.
70. Meirelles P, Santos C, Miranda J, et al. A study of the relationships between source code metrics and attractiveness in free software projects. In: *Proceedings of the Brazilian symposium on software engineering (SBES)*, 2012, pp. 11–20, http://flossmole.org
71. Lanza M and Marinescu R. *Object-oriented metrics in practice*. Berlin: Springer, 2006.