SIDE: A Web App for Interactive Visual Data Exploration with Subjective Feedback

Jefrey Lijffijt[1] Bo Kang[1] Kai Puolamäki[2] Tijl De Bie[1]
[1] Data Science Lab, Ghent University, Belgium
[2] Finnish Institute of Occupational Health, Finland
{jefrey.lijffijt;bo.kang;tijl.debie}@ugent.be, kai.puolamaki@ttl.fi

ABSTRACT

Data visualization and iterative/interactive data mining are growing rapidly in attention, both in research as well as in industry. However, integrated methods and tools that combine advanced visualization and/or interaction with data mining techniques are rare, and those that exist are specialized to a single problem or domain. We present SIDE, a generic tool for Subjective Interactive Data Exploration, which lets users explore high dimensional data via subjectively informative two-dimensional data visualizations. In contrast to most visualization tools, it is not based on the traditional dogma of manually zooming and rotating data. Instead, the tool initially presents the user with an 'interesting' projection, and then allows users to flexibly and intuitively express their interests or beliefs using visual interactions that update/constrain a background model of the data. These constraints expressed by the user are then taken into account by a projection-finding algorithm employing data randomization to compute a new 'interesting' projection. This process can be iterated until the user runs out of time or finds that the difference between the randomized data and the real data is no longer interesting. We present the tool by means of two case studies, one controlled study on synthetic data and another on real census data.

Keywords

Exploratory Data Mining; Dimensionality Reduction; Data Randomization; Subjective Interestingness

1. INTRODUCTION

Data visualization and iterative/interactive data mining are both mature, actively researched topics of great practical importance. However, while progress in both fields is abundant (see Section 4), methods that combine iterative data mining with visualization and interaction are rare; only a few tools designed for specific problem domains exist.

Yet, tools that combine state-of-the-art data mining with visualization and interaction are highly desirable as they would maximally exploit the strengths of both human data analysts and computer algorithms. Humans are unmatched in spotting interesting patterns in low-dimensional visual representations, but poor at reading high-dimensional data, while computers excel in manipulating high-dimensional data and are weaker at identifying patterns that are truly relevant to the user. A symbiosis of human analysts and well-designed computer systems thus promises to provide an efficient way of navigating the complex information space hidden within high-dimensional data [17].

Contributions.

In this paper we introduce a generically applicable method for finding interesting projections of data, given some prior knowledge about that data, and we introduce a tool that demonstrates the proposed approach for interactive visual exploration of (high-dimensional) data. The underlying idea is that the analysis process is iterative, and during each iteration there are three steps. The hypothesis is that throughout the iterations, the user builds up an increasingly accurate understanding of the data. This understanding is explicated in the *background model*, which is used at the beginning of each iteration in order to find a maximally informative projection. More generally, the background model is a representation for the user's *belief state.* The tool works as indicated in Figure 1. Details of all steps are given below.

Step 1. The tool initially presents the user with an 'interesting' projection of the data, visualized as a scatter plot. Here, interestingness is formalized with respect to the initial belief state.

Step 2. On investigation of this scatter plot, the user may take note of some features of the data that contrast with, or add to, their beliefs about the data. We will refer to such features as *patterns.* The user then interacts with the tool

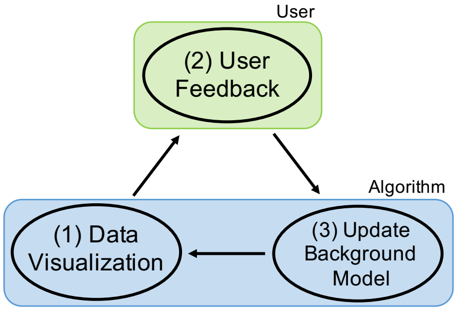


Figure 1: The three steps of SIDE's operation cycle.

IDEA Workshop, SIGKDD '16, August 14, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN –. . . $0.00
DOI: --

to indicate what patterns they have seen and assimilated.
Step 3. The tool updates the background model according to the user feedback, in order to reflect the newly assimilated information.
Next iteration. Then the most interesting projection with respect to this updated background model can be computed, and the cyclic process iterates until the user runs out of time or finds that background model (and thus the user's belief state) explains everything the user is currently interested in.

Formalization of the background model.

A crucial challenge in the realization of such a tool is the formalization of the background model. To allow the process to be iterative, the formalization has to allow for the model to be updated after a user has been provided with new information (i.e., shown a visualization) and given feedback on it. There exist two frameworks for iterative data mining: FORSIED [3, 4] and a framework that has no name yet, but which we will refer to as CORAND [7, 13], for COnstrained RANDomization. In both cases, the background model is a probability distribution over data sets and the user beliefs are modelled as a set of *constraints* on that distribution.

The CORAND approach is to specify a randomization procedure that, when applied to the data, does not affect how plausible the user would deem it to be. That is, the user's beliefs should be satisfied, and otherwise the data should be shuffled as much as possible. Given an appropriate randomization scheme, we can then find interesting remaining structure that is not yet known to the user by contrasting the real data against the randomized data. New beliefs can be incorporated in the background model by adding corresponding constraints to the randomization procedure, ensuring that the patterns observed by the user are present also in the subsequent randomized data.

An illustrative example.

As an example, consider a synthetic data set that consists of 1000 ten-dimensional data vectors of which dimensions 1–4 can be clustered into five clusters, dimensions 5–6 into four clusters *involving different subsets of data points*, and of which dimensions 7–10 are Gaussian noise. All dimensions have equal variance.

We designed this example to illustrate the two types of feedback that a user can give in the current implementation of our tool. Additionally, it shows how the tool succeeds in finding interesting projections given previously identified patterns. Thirdly, it also demonstrates how the user interactions meaningfully affect subsequent visualizations. In this example we aim to provide an overview of how the tool works, technical details are presented in Section 2.

We observe that the first projection computed by SIDE maps the data onto a two-dimensional (2D) subspace of the dimensions 1–4 (Figure 2a), i.e., to a subspace of the space where the data is clustered into 5 clusters. This is indeed sensible, as the structure within this 4D subspace is arguably the most striking.

We then consider two possible user actions (Step 2, Figure 2b). In the first scenario (Figure 2 left path), the user marks all points within each cluster (one cluster at a time), indicating they have taken note of the positions of these groups of points *within this particular projection.* In the second scenario (Figure 2 right path), the user gives the feedback that these points appear to be clustered *in this projection and possibly also in other dimensions.*

Both these 'pattern types' lead to a set of constraints on the randomization procedure. The effect of these constraints is identical with respect to the current 2D projection (Figure 2c): the projections of the randomized points onto this plane are identical to the projections of the original points onto this plane. Not visible though is that in the second scenario the randomization is restricted also in orthogonal dimensions (possibly different ones for different clusters), to account for the user feedback that also orthogonal subspaces that yield the same clusters are not interesting anymore.

The subsequent most interesting projection is different in the two scenarios (Figure 2d). In the first scenario, the remaining cluster structure within dimensions 1–4 is shown. However, in the second scenario this cluster structure is fully explained by the constraints, and as a result, the cluster structure in dimensions 5–6 being is shown instead.

The difference can be observed in the visualization because on the left three clusters are pure and one is mixed (an artefact of how we chose the cluster centers). Yet, on the right all clusters are mixed with respect the previous clustering. This indeed shows the two clusterings in dimensions 1–4 and dimensions 5–6 are unrelated.

Outline of this paper.

As discussed in Section 2, three challenges had to be addressed to use the CORAND approach: (1) defining intuitive pattern types (constraints) that can be observed and specified based on a scatter plot of a two-dimensional projection of the data; (2) defining a suitable randomization scheme, that can be constrained to take account of such patterns; and (3) a way to identify the most interesting projections given the background model. The evaluation with respect to usefulness as well as computational properties of the resulting system is presented in Section 3. Experiments were conducted both on synthetic data and on a census dataset. Finally, related work and conclusions are discussed in Sections 4 and 5, respectively.

NB. This manuscript is an integration of two publications that are to appear in the Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery [10, 16].

2. METHODS

We will use the notational convention that upper case bold face symbols represent matrices, lower case bold face symbols represent column vectors, and lower case standard face symbols represent scalars. We assume that our data set consists of n d-dimensional data vectors $\mathbf{x}_i$. The data set is represented by a real matrix $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T & \mathbf{x}_2^T & \cdots & \mathbf{x}_n^T \end{pmatrix}^T \in \mathbb{R}^{n \times d}$. More generally, we will denote the transpose of the ith row of any matrix $\mathbf{A}$ as $\mathbf{a}_i$ (i.e., $\mathbf{a}_i$ is a column vector). Finally, we will use the shorthand notation $[n] = \{1, \ldots, n\}$.

2.1 Projection tile patterns in two flavours

In the interaction step, the proposed system allows users to declare that they have become aware of (and thus are no longer interested in seeing) the value of the projections of a set of points onto a specific subspace of the data space. We call such information a *projection tile* pattern for reasons that will become clear later. A projection tile parametrizes a set of constraints to the randomization.

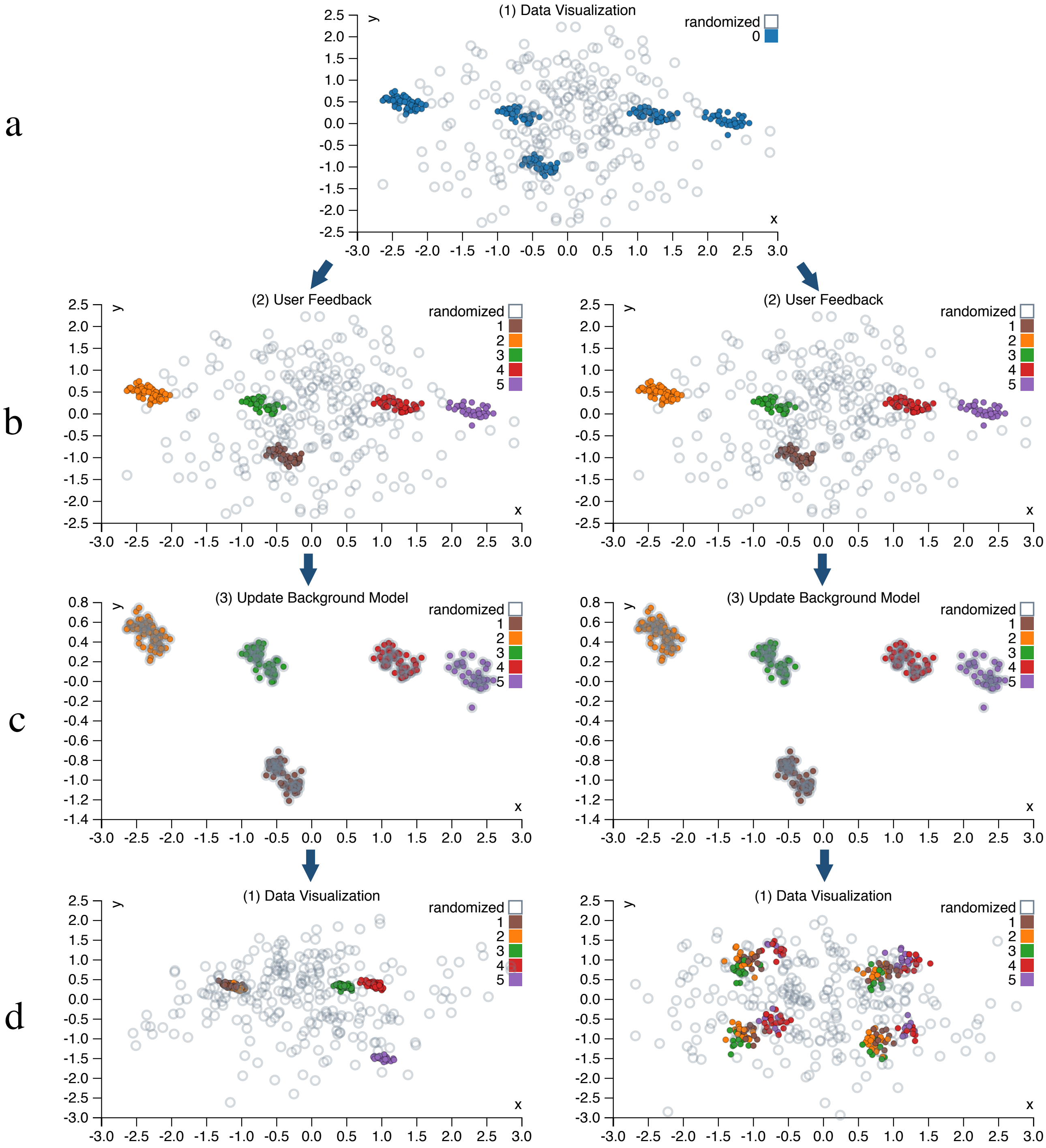


Figure 2: Two user interaction scenarios for the toy data set. Solid dots represent actual data vectors, whereas open circles represent vectors from the randomized data. Row (a) shows the first visualization, which is the starting point for both scenarios. Row (b) shows the sets of data points marked by the user. Although not shown, on the left the user gives feedback to incorporate the selected cluster structure in the currently shown dimensions, while on the right the feedback is that the user expects the cluster structure to generalize to other unshown dimensions. Row (c) shows the newly randomized data and the original data projected still in the same subspace. As expected, the randomized data fully aligns with the real data. Then, row (d) shows the most interesting visualization given the specified patterns (constraints). The left path shows the scenario when the user assumes nothing beyond the values of the data points in the projection in row (a), whereas the right path shows the scenario when the user assumes each of these sets of points may be clustered in other dimensions as well.

Formally, a projection tile pattern, denoted τ, is defined by a k-dimensional (with $k \leq d$ and $k = 2$ in the simplest case) subspace of $\mathbb{R}^d$, and a subset of data points $\mathcal{I}_\tau \subseteq [n]$. We will formalize the k-dimensional subspace as the column space of an orthonormal matrix $\mathbf{W}_\tau \in \mathbb{R}^{d \times k}$ with $\mathbf{W}_\tau^T \mathbf{W}_\tau = \mathbf{I}$, and can thus denote the projection tile as $\tau = (\mathbf{W}_\tau, \mathcal{I}_\tau)$. The proposed tool provides two ways in which the user can define the projection vectors $\mathbf{W}_\tau$ for a projection tile τ.

2D tiles.

The first approach simply chooses $\mathbf{W}_\tau$ as the two weight vectors defining the projection within which the data vectors belonging to $\mathcal{I}_\tau$ were marked. This approach allows the user to simply specify that they have taken note of the positions of that set of data points within this projection. The user makes no further assumptions—they assimilate solely what they see without drawing conclusions not supported by direct evidence, see Figure 2b (left).

Clustering tiles.

It seems plausible, however, that when the marked points are tightly clustered, the user concludes that these points are clustered *not just within the two dimensions shown* in the scatter plot. To allow the user to express such belief, the second approach takes $\mathbf{W}_\tau$ to additionally include a basis for other dimensions along which these data points are strongly clustered, see Figure 2b (right). This is achieved as follows.

Let $\mathbf{X}(\mathcal{I}_\tau, :)$ represent a matrix containing the rows indexed by elements from $\mathcal{I}_\tau$ from $\mathbf{X}$. Let $\mathbf{W} \in \mathbb{R}^{d \times 2}$ contain the two weight vectors onto which the data was projected for the current scatter plot. In addition to $\mathbf{W}$, we want to find any other dimensions along which these data vectors are clustered. These dimensions can be found as those along which the variance of these data points is not much larger than the variance of the projection $\mathbf{X}(\mathcal{I}_\tau, :)\mathbf{W}$.

To find these dimensions, we first project the data onto the subspace orthogonal to $\mathbf{W}$. Let us represent this subspace by a matrix with orthonormal columns, further denoted as $\mathbf{W}^\perp$. Thus, ${\mathbf{W}^\perp}^T \mathbf{W}^\perp = \mathbf{I}$ and $\mathbf{W}^T \mathbf{W}^\perp = \mathbf{0}$. Then, Principal Component Analysis (PCA) is applied to the resulting matrix $\mathbf{X}(\mathcal{I}_\tau, :)\mathbf{W}^\perp$. The principal directions corresponding to a variance smaller than a threshold are then selected and stored as columns in a matrix $\mathbf{V}$. In other words, the variance of each of the columns of $\mathbf{X}(\mathcal{I}_\tau, :)\mathbf{W}^\perp\mathbf{V}$ is below the threshold.

The matrix $\mathbf{W}_\tau$ associated to the projection tile pattern is then taken to be:

$$\mathbf{W}_\tau = \begin{pmatrix} \mathbf{W} & \mathbf{W}^\perp \mathbf{V} \end{pmatrix}.$$

The threshold on the variance used could be a tunable parameter, but was set here to twice the average of the variance of the two dimensions of $\mathbf{X}(\mathcal{I}_\tau, :)\mathbf{W}$.

2.2 The randomization procedure

Here we describe the approach to randomizing the data. The randomized data should represent a sample from an implicitly defined background model that represents the user's belief state about the data. Initially, our approach assumes the user merely has an idea about the overall scale of the data. However, throughout the interactive exploration, the patterns in the data described by the projection tiles will be maintained in the randomization.

Initial randomization.

The proposed randomization procedure is parametrized by n orthogonal rotation matrices $\mathbf{U}_i \in \mathbb{R}^{d \times d}$, where $i \in [n]$, and the matrices satisfy $(\mathbf{U}_i)^T = (\mathbf{U}_i)^{-1}$. We further assume that we have a bijective mapping $f : [n] \times [d] \mapsto [n] \times [d]$ that can be used to permute the indices of the data matrix. The randomization proceeds in three steps:

Random rotation of the rows Each data vector $\mathbf{x}_i$ is rotated by multiplication with its corresponding random rotation matrix $\mathbf{U}_i$, leading to a randomised matrix $\mathbf{Y}$ with rows $\mathbf{y}_i^T$ that are defined by:

$$\forall i : \ \mathbf{y}_i = \mathbf{U}_i \mathbf{x}_i.$$

Global permutation The matrix $\mathbf{Y}$ is further randomized by randomly permuting all its elements, leading to the matrix $\mathbf{Z}$ defined as:

$$\forall i, j : \ \mathbf{Z}_{i,j} = \mathbf{Y}_{f(i,j)}.$$

Inverse rotation of the rows Each randomised data vector in $\mathbf{Z}$ is rotated with the inverse rotation applied in step 1, leading to the fully randomised matrix $\mathbf{X}^*$ with rows $\mathbf{x}_i^*$ defined as follows in terms of the rows $\mathbf{z}_i^T$ of $\mathbf{Z}$:

$$\forall i : \ \mathbf{x}_i^* = {\mathbf{U}_i}^T \mathbf{z}_i.$$

The random rotations $\mathbf{U}_i$ and the permutation f are sampled uniformly at random from all possible rotation matrices and permutations, respectively.

Intuitively, this randomization scheme preserves the scale of the data points. Indeed, the random rotations leave their lengths unchanged, and the global permutation subsequently shuffles the values of the d components of the rotated data points. Note that without the permutation step, the two rotation steps would undo each other such that $\mathbf{X}^* = \mathbf{X}$. Thus, it is the combined effect that results in a randomization of the data set.

The random rotations may seem superfluous: the global permutation randomizes the data so dramatically that the added effect of the rotations is relatively unimportant. However, their role is to make it possible to formalize the growing understanding of the user as simple constraints on this randomization procedure, as discussed next.

Accounting for one projection tile.

Once the user has assimilated the information in a projection tile $\tau = (\mathbf{W}_\tau, \mathcal{I}_\tau)$, the randomization scheme should incorporate this information by ensuring that it is present also in all randomized versions of the data. This ensures that the randomized data is a sample from a distribution representing the user's belief state about the data. This is achieved by imposing the following *constraints* on the parameters defining the randomization:

Rotation matrix constraints For each $i \in \mathcal{I}_\tau$, the component of $\mathbf{x}_i$ that is within the column space of $\mathbf{W}_\tau$ must be mapped onto the first k dimensions of $\mathbf{y}_i = \mathbf{U}_i \mathbf{x}_i$ by the rotation matrix $\mathbf{U}_i$. This can be achieved by ensuring that:

$$\forall i \in \mathcal{I}_\tau : \ \mathbf{W}_\tau^T \mathbf{U}_i = \begin{pmatrix} \mathbf{I} & \mathbf{0} \end{pmatrix}. \quad (1)$$

This explains the name *projection tile*: the information to be preserved in the randomization is concentrated in a 'tile' (i.e. the intersection of a set of rows and a set of columns) in the intermediate matrix $\mathbf{Y}$ created during the randomization procedure.

Permutation constraints The permutation should not affect any matrix cells with row indices $i \in \mathcal{I}_\tau$ and columns indices $j \in [k]$:

$$\forall i \in \mathcal{I}_\tau, j \in [k]: \ f(i,j) = (i,j). \qquad (2)$$

PROPOSITION 1. *Using the above constraints on the rotation matrices $\mathbf{U}_i$ and the permutation f, it holds that:*

$$\forall i \in \mathcal{I}_\tau, \mathbf{x}_i^T \mathbf{W}_\tau = \mathbf{x}_i^{*T} \mathbf{W}_\tau. \qquad (3)$$

Thus, the values of the projections of the points in the projection tile remain unaltered by the constrained randomization. Hence, the randomization keeps the user's beliefs intact. We omit the proof as the more general Proposition 2 is provided with proof further below.

Accounting for multiple projection tiles.

Throughout subsequent iterations, additional projection tile patterns will be specified by the user. A set of tiles τ_i for which $\mathcal{I}_{\tau_i} \cap \mathcal{I}_{\tau_j} = \emptyset$ if $i \neq j$ is straightforwardly combined by applying the relevant constraints on the rotation matrices to the respective rows. When the sets of data points affected by the projection tiles overlap though, the constraints on the rotation matrices need to be combined. The aim of such a combined constraint should be to preserve the values of the projections onto the projection directions for *each* of the projection tiles a data vector was part of.

The combined effect of a set of tiles will thus be that the constraint on the rotation matrix $\mathbf{U}_i$ will vary per data vector, and depends on the set of projections $\mathbf{W}_\tau$ for which $i \in \mathcal{I}_\tau$. More specifically, we propose to use the following constraint on the rotation matrices:

Rotation matrix constraints Let $\mathbf{W}_i \in \mathbb{R}^{d \times d_i}$ denote a matrix of which the columns are an orthonormal basis for space spanned by the union of the columns of the matrices $\mathbf{W}_\tau$ for τ with $i \in \mathcal{I}_\tau$. Thus, for any i and $\tau : i \in \mathcal{I}_\tau$, it holds that $\mathbf{W}_\tau = \mathbf{W}_i \mathbf{v}_\tau$ for some $\mathbf{v}_\tau \in \mathbb{R}^{d_i}$. Then, for each data vector i, the rotation matrix $\mathbf{U}_i$ must satisfy:

$$\forall i \in \mathcal{I}_\tau : \ \mathbf{W}_i^T \mathbf{U}_i = (\mathbf{I} \ \ \mathbf{0}). \qquad (4)$$

Permutation constraints Then the permutation should not affect any matrix cells in row i and columns $[d_i]$:

$$\forall i \in [n], j \in [d_i]: \ f(i,j) = (i,j).$$

PROPOSITION 2. *Using the above constraints on the rotation matrices $\mathbf{U}_i$ and the permutation f, it holds that:*

$$\forall \tau, \forall i \in \mathcal{I}_\tau, \mathbf{x}_i^T \mathbf{W}_\tau = \mathbf{x}_i^{*T} \mathbf{W}_\tau.$$

PROOF. We first show that $\mathbf{x}_i^{*T} \mathbf{W}_i = \mathbf{x}_i^T \mathbf{W}_i$:

$$\mathbf{x}_i^{*T} \mathbf{W}_i = \mathbf{z}_i^T \mathbf{U}_i^T \mathbf{W}_i = \mathbf{z}_i^T \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix}$$

$$= \mathbf{z}_i(1:d_i)^T = \mathbf{y}_i(1:d_i)^T = \mathbf{y}_i^T \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix} = \mathbf{x}_i^T \mathbf{W}_i.$$

The result now follows from the fact that $\mathbf{W}_\tau = \mathbf{W}_i \mathbf{v}_\tau$ for some $\mathbf{v}_\tau \in \mathbb{R}^{d_i}$. □

Technical implementation of the randomization.

To ensure the randomization can be carried out efficiently throughout the process, note that the matrix $\mathbf{W}_i$ for the $i \in \mathcal{I}_\tau$ for a new projection tile τ can be updated by computing an orthonormal basis for $(\mathbf{W}_i \ \ \mathbf{W})$. Such a basis can be found efficiently as the columns of $\mathbf{W}_i$ in addition to the columns of an orthonormal basis of $\mathbf{W} - \mathbf{W}_i^T \mathbf{W}_i \mathbf{W}$ (the components of $\mathbf{W}$ orthogonal to $\mathbf{W}_i$), the latter of which can be computed using the QR-decomposition.

Additionally, note that the tiles define an equivalence relation over the row indices, in which i and j are equivalent if they were included in the same set of projection tiles so far. Within each equivalence class, the matrix $\mathbf{W}_i$ will be constant, such that it suffices to compute it only once, keeping track of which points belong to which equivalence class.

2.3 Visualization: Finding the most interesting two-dimensional projection

Given the data set $\mathbf{X}$ and the randomized data set $\mathbf{X}^*$, it is now possible to quantify the extent to which the empirical distribution of a projection $\mathbf{Xw}$ and $\mathbf{X}^*\mathbf{w}$ onto a weight vector $\mathbf{w}$ differ. There are various ways in which this difference can be quantified. We investigated a number of possibilities and found that the L_1-distance between the cumulative distribution functions works well in practice. Thus, with $F_\mathbf{x}$ the empirical cumulative distribution function for the set of values in $\mathbf{x}$, the optimal projection is found by solving:

$$\max_{\mathbf{w}} \left\| F_{\mathbf{Xw}} - F_{\mathbf{X}^*\mathbf{w}} \right\|_1 .$$

The second dimension of the scatter plot can be sought by optimizing the same objective while requiring it to be orthogonal to the first dimension.

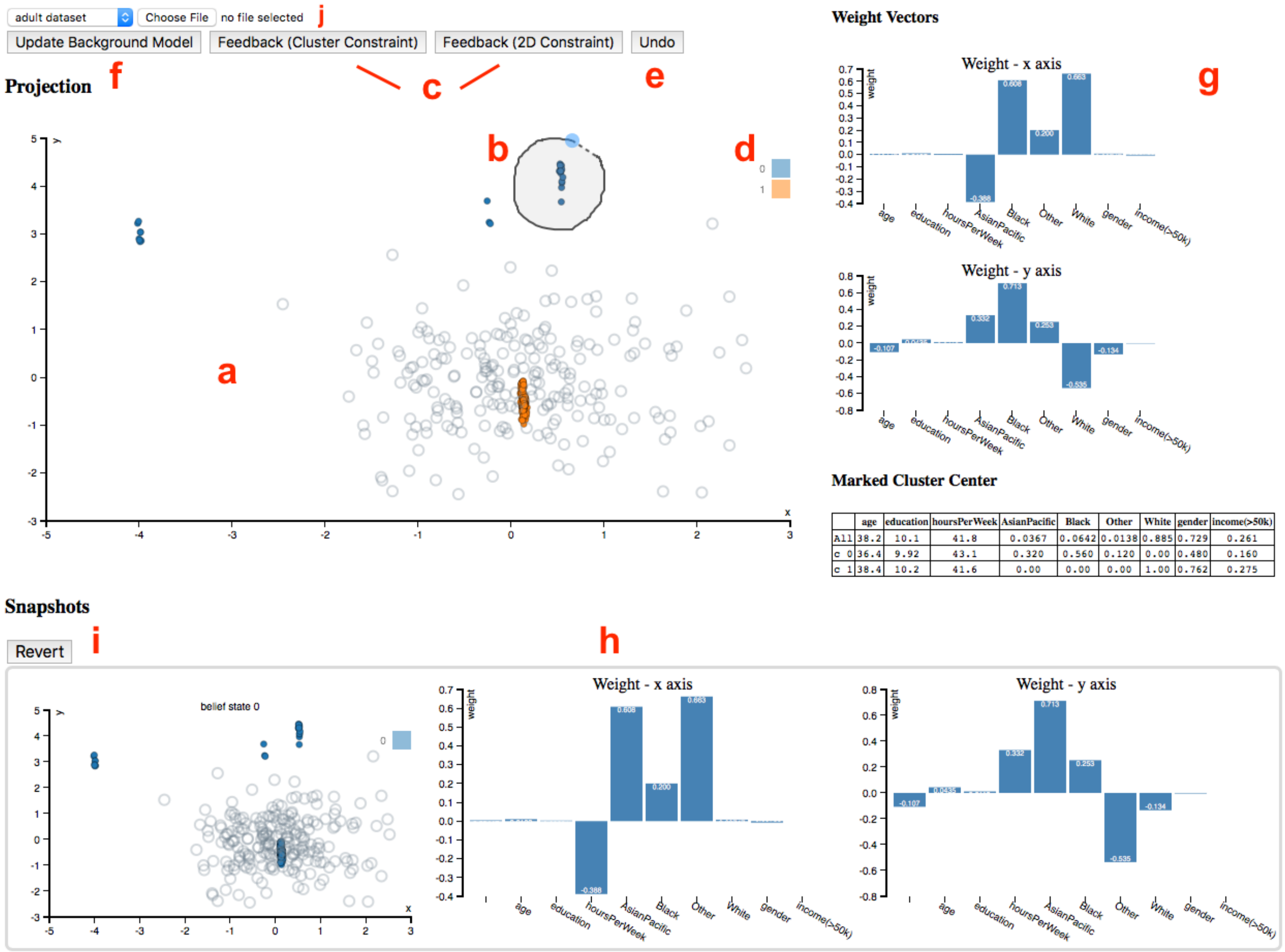
We are unaware of any special structure of this optimization problem that makes solving it particularly efficient. Yet, using the standard quasi-Newton solver in R [18] with random initialization and default settings (the general-purpose `optim` function with `method`="BFGS") already yields satisfactory results, as shown in the experiments below.

2.4 Interface

The full interface of SIDE is shown in Figure 3. SIDE was designed according to three principles for visually controllable data mining [17], which essentially says that both the model and the interactions should be transparent to users, and that the analysis method should be fast enough such that the user does not lose its trail of thought.

The main component is the interactive scatter plot (3a). The scatter plot visualizes the projected data (solid dots) and the randomized data (open gray circles) in the current 2D projection. By drawing circles (3b), the user can highlight data points to define a *projection tile pattern.* Once a set of points is marked, the user can press either of the two feedback buttons (3c), to indicate these points form a cluster. If the user thinks the points are clustered only in the shown projection, they click '2D Constraint', while 'Cluster Constraint' indicates they expect that these points will be clustered in other dimensions as well.

To identify the defined clusters, data points associated with the same feedback (i.e., user's belief) are filled by the same color (3d), and their statistics are shown in a table. The user can define multiple clusters in a single projection, and they can also *undo* (3e) the feedback. Once a user finishes exploring the current projection, they can press 'Up-



Marked Cluster Center

	age	education	hoursPerWeek	AsianPacific	Black	Other	White	gender	income(>50k)
All	38.2	10.1	41.8	0.0367	0.0642	0.0138	0.885	0.729	0.261
c 0	36.4	9.92	43.1	0.320	0.560	0.120	0.00	0.480	0.160
c 1	38.4	10.2	41.6	0.00	0.00	0.00	1.00	0.762	0.275

Figure 3: Layout of our web app SIDE, which contains the data visualization and interaction area (a–f), projection meta information (g), and timeline (h).

date Background Model' (3f). Then, the background model is updated with the provided feedback and a new scatter plot is computed and presented to the user, etc.

A few extra features are provided to assist the data exploration process: to gain an understanding of a projection, the weight vectors associated with the projection axes are plotted as bar charts (3g). At the bottom of 3g, a table lists the mean vectors of each colored point set (i.e., cluster). The exploration history is maintained by taking snapshots of the background model when updated, together with the associated data projection (scatter plot) and bar charts (weight vectors). This history in reverse chronological order is illustrated in Figure 3h.

The tool also allows a user to click and revert back to a certain snapshot (3i), to restart from that time point. This allows the user to discover different aspects of a dataset more consistently. Finally, custom datasets can be selected for analysis from the drop-down menu (3j). Currently our tool only works with CSV files and it automatically sub-samples the custom data set so that the interactive experience is not compromised. By default, two datasets are preloaded so that users can get familiar with the tool.

3. EXPERIMENTS

We present two case studies to illustrate the framework and its utility. The case studies are completed with the a JavaScript version of our tool, which is available freely online, along with the used data for reproducibility.[1]

3.1 Synthetic data case study

This section gives an extended discussion of the illustrative example from the introduction, namely the synthetic data case study. The data is described in Section 1. The first projection shows that the projected data (solid blue dots in Figure 2a) differs strongly from the randomized data (open gray circles). The weight vectors defining the projection, shown in the 1st row of Table 1, contain large weights in dimensions 1–4. Therefore, the cluster structure seen here mainly corresponds to dimensions 1–4 of the data.

A user can indicate this insight by means of a *clustering tile* for each of the clustered sets of data points (2b, right). Encoding this into the background model, results in a randomization, where the randomized points perfectly

[1] http://www.interesting-patterns.net/forsied/a-tool-for-subjective-and-interactive-visual-data-exploration/)

Table 1: Projection weight vectors for the synthetic data (Sections 1 and 3.1).

Figure	axis	1	2	3	4	5	6	7	8	9	10
2a	X	**0.194**	**0.545**	**-0.630**	**0.499**	-0.119	-0.041	0.057	0.001	-0.029	0.003
	Y	**-0.269**	**-0.754**	**-0.481**	**0.340**	0.091	-0.004	0.016	-0.057	0.003	0.005
2d	X	**0.143**	**-0.118**	0.005	**0.981**	0.001	-0.013	-0.031	-0.022	0.044	-0.031
(left)	Y	**-0.245**	**0.448**	**0.854**	0.088	0.004	-0.001	0.005	0.008	-0.043	0.023
2d	X	0.121	0.019	-0.232	0.017	**-0.963**	-0.008	0.022	0.023	0.037	0.004
(right)	Y	-0.139	-0.067	-0.369	-0.082	0.111	**-0.898**	-0.083	0.086	0.005	-0.017

Table 2: Projection weight vectors for the UCI Adult data (Section 3.2).

Figure	axis	Age	Edu.	h/w	EG_AsPl	EG_Bl.	EG_Oth.	EG_Whi.	Gender	Income
4a	X	-0.039	-0.001	0.001	**0.312**	**-0.530**	**-0.193**	**0.763**	0.017	0.008
	Y	0.004	-0.004	-0.002	**0.816**	**-0.141**	**0.465**	**-0.313**	-0.011	0.002
4c	X	0.081	-0.028	-0.022	-0.259	-0.233	-0.104	-0.380	**-0.846**	-0.001
	Y	-0.590	0.541	0.143	-0.233	-0.380	-0.026	-0.293	0.232	0.000
4d	X	0.119	-0.149	0.047	0.102	0.191	0.104	**-0.556**	0.0581	**-0.769**
	Y	**-0.382**	**-0.626**	**-0.406**	0.346	0.317	-0.0287	0.111	-0.248	0.059

Table 3: Mean vectors of user marked clusters for the UCI Adult data (Section 3.2).

Figure	Cluster	Age	Edu.	h/w	EG_AsPl	EG_Bl.	EG_Oth.	EG_Whi.	Gender	Income
4b	top left	35.0	8.67	34.7	0.00	0.00	**1.00**	0.00	0.667	0.333
	bott. left	37.2	9.43	40.3	0.00	**1.00**	0.00	0.00	0.286	0.071
	top right	35.6	1.3	51.1	**1.00**	0.00	0.00	0.00	0.750	0.250
	bott. right	38.4	10.2	41.6	0.00	0.00	0.00	**1.00**	0.762	0.275
4c	left	39.0	10.2	43.3	0.0377	0.0252	0.0126	0.925	**1.00**	0.321
	right	36.0	9.95	37.9	0.0339	0.169	0.0169	0.780	**0.00**	0.102
4d	left	**42.5**	**11.6**	**46.3**	0.00	0.00	0.00	**1.00**	1.00	**1.00**

align with data points (2c, right). The new projection that differs most from this updated background model reveals the four clusters in dimensions 5–6 that the user was not aware of before (2d, right).

If the user does not want to draw conclusions about the points being clustered in dimensions other than those shown, she can use *2D tiles* instead of *clustering tiles* (Figure 2b, left). The updated background model then results in a randomization that is indistinguishable in the given projection from the one with a clustering tile (2c, left), but it results in a different subsequent projection (2d, left). Indeed, this leads to just another view of the five clusters in dimensions 1–4, as confirmed by the large weights for dimensions 1–4 (2nd row of Table 1). Thus, by these simple interactions the user can choose whether she will allow additional exploration of the cluster structure in dimensions 1–4 or if she is now already aware of the cluster structure, in which case the system directs her to the structure occurring in dimensions 5–6. This behavior aligns perfectly with our expectations.

3.2 UCI Adult dataset case study

In this case study, we demonstrate the utility of our method by exploring a real world dataset. The data is compiled from UCI Adult dataset[2]. To ensure the real time interactivity, we sub-sampled 218 data points and selected six features: “Age” (17 – 90), “Education” (1 – 16), “HoursPerWeek” (1 – 99), “Ethnic Group” (White, AsianPacIslander, Black, Other), “Gender” (Female, Male), “Income” ($\geq 50k$). Among the selected features, “Ethnic Group” is a categorical feature with five categories, “Gender” and “Income” are binary features, the rest are all numeric. To make our method applicable to this dataset, we further binarized the “Ethnic Group” feature (yielding four binary features), and the final dataset consists of 218 points and 9 features.

We assume the user uses clustering tiles throughout the exploration. Each of the patterns discovered during the exploration process thus corresponds to a certain demographic clustering pattern. To illustrate how our tool helps the user rapidly gain an understanding of the data, we discuss the first three iterations of the exploration process. The first projection (Figure 4a) visually consists of four clusters. The user notes that the weight vectors corresponding to the axes of the plot assign large weights to the “Ethnic Group” attributes (Table 2, 1st row). As mentioned, we assume the user marks these points as part of the same clustering tile. When marking the clusters (Figure 4b), the tool informs the user of the mean vectors of the points within each clustering tile. The 1st row of Table 3 shows that each cluster completely represents one out of four ethnic groups, which may corroborate with the user’s understanding.

Taking the user’s feedback into consideration, a new projection is generated by the tool. The new scatter plot (Figure 4c) shows two large clusters, each consisting of some points from the previous four-cluster structure (points from these four clusters are colored differently). Thus, the new scatter plot elucidates structure not shown in the previous one. Indeed, the weight vectors (2nd row of Table 2) show that the clusters are separated mainly according to the “Gender” attribute. After marking the two clusters separately, the mean vector of each cluster (2nd row of Table 3) again confirms this: the cluster on the left represents male group, and the female group is on the right.

[2] https://archive.ics.uci.edu/ml/datasets/Adult

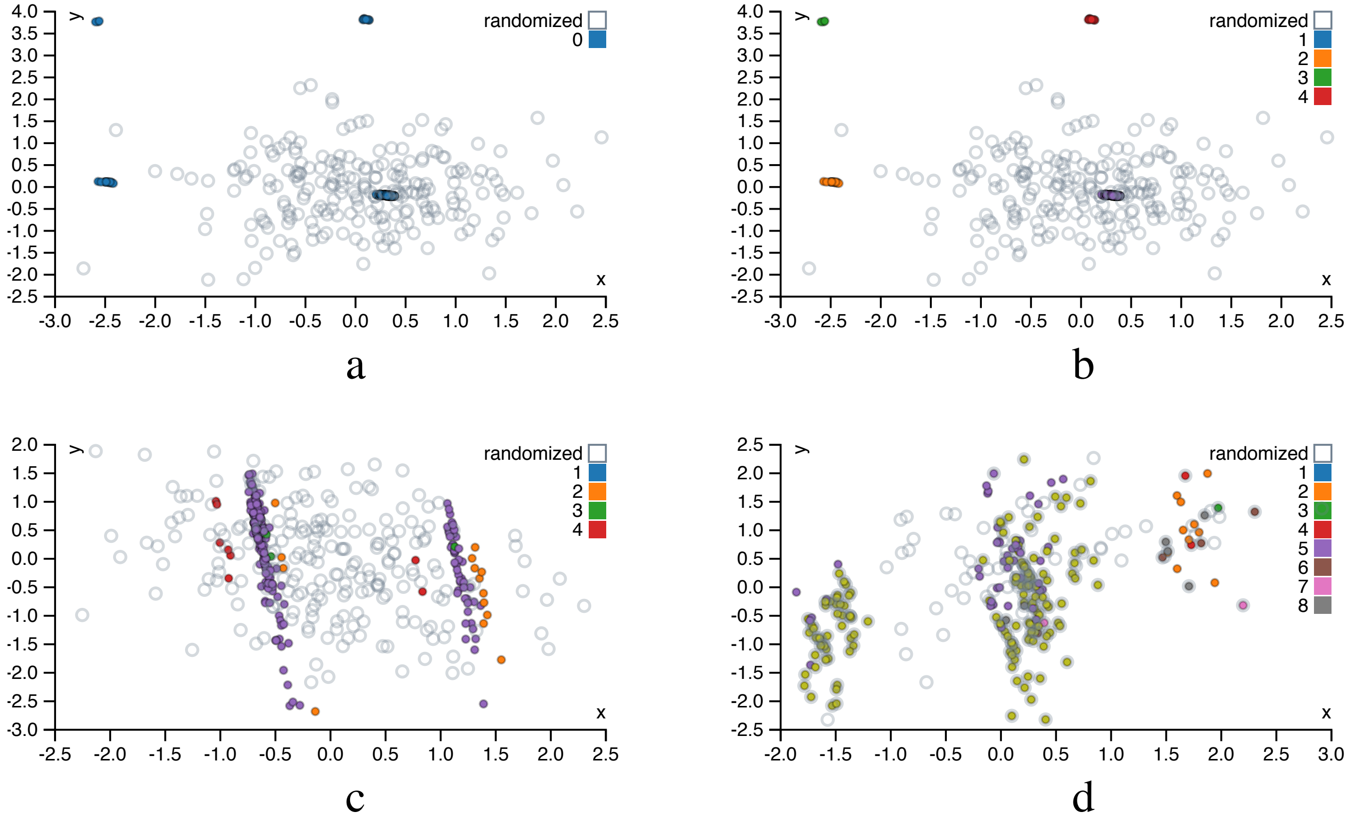


Figure 4: Projections of UCI Adult dataset: (a) projection in the $1st$ iteration, (b) clusters marked by user in the $1st$ iteration, (c) projection in the $2nd$ iteration, and (d) projection in the $3rd$ iteration

The projection in the third iteration (Figure 4d) consists of three clusters, separated only along the x-axis. Interestingly, the corresponding weight vector (3rd row of Table 2) has strongly negative weights for the attributes "Income" and "Ethnic Group - White". This indicates the left cluster mainly represents the people with high income and whose ethnic group is also "White". This cluster has relatively low y-value; according to the weight vector, they are also generally older and more highly educated. These observations are corroborated by the cluster mean (Table 3, 3rd row).

This case study illustrates how the proposed tool facilitates human data exploration by iteratively presenting an informative projection, considering what the user has already learned about the data.

3.3 Performance on synthetic data

Ideally interactive data exploration tools should work in close to real time. This section contains an empirical analysis of an (unoptimized) R implementation of our tool, as a function of the size, dimensionality, and complexity of the data. Note that limits on screen resolution as well as on human visual perception render it useless to display more than of the order of a few hundred data vectors, such that larger data sets can be down-sampled without noticeably affecting the content of the visualizations.

We evaluated the scalability on synthetic data with $d \in \{16, 32, 64, 128\}$ dimensions and $n \in \{64, 128, 256, 512\}$ data points scattered around $k \in \{2, 4, 8, 16\}$ randomly drawn cluster centroids (Table 4). The randomization is done here with the initial background model. The most costly part in randomization is usually the multiplication of orthogonal matrices, indeed, the running time of the randomization scales roughly as nd^{2-3}. The results suggests that the running time of the optimization is roughly proportional to the size of the data matrix nd and that the complexity of data k has here only a minimal effect in the running time of the optimization.

Furthermore, in 90% of the tests, the L_1 loss on the first axis is within 1% of the best L_1 norm out of ten restarts. The optimization algorithm is therefore quite stable, and in practical applications it may well be be sufficient to run the optimization algorithm only once. These results have been obtained with unoptimized and single-threaded R implementation on a laptop having 1.7 GHz Intel Core i7 processor.[3] The performance could probably be significantly boosted by, e.g., carefully optimizing the code and the implementation. Yet, even with this unoptimized code, response times are already of the order of 1 second to 1 minute.

4. RELATED WORK

Dimensionality reduction.

Dimensionality reduction for exploratory data analysis has been studied for decades. Early research into visual exploration of data led to approaches such as multidimensional

[3] The R implementation used to produce Table 4 is available also via the demo page (footnote 1).

Table 4: Median wall clock running times, for randomization and optimization over ten iterations of finding 2D-projections using L_1 loss. Also shown is the number of iterations in which the L_1 norm first component ended up within 1% of the result with the largest L_1 norm (out of 10 tries). A high number indicates the solution quality is stable, even though the actual projections may vary.

		rand.	$k \in \{2, 4, 8, 16\}$	
n	d	(s)	optim. (s)	#tries $\Delta < 1\%$
64	16	0.1	$\{1.0, 1.2, 0.9, 1.2\}$	$\{10, 10, 9, 8\}$
64	32	0.5	$\{1.8, 2.1, 2.4, 2.5\}$	$\{10, 8, 10, 10\}$
64	64	2.5	$\{5.6, 3.5, 4.6, 4.5\}$	$\{10, 9, 10, 8\}$
64	128	11.5	$\{8.9, 10.1, 11.4, 10.2\}$	$\{10, 10, 8, 9\}$
128	16	0.2	$\{2.0, 1.7, 2.4, 2.0\}$	$\{10, 1, 6, 8\}$
128	32	0.8	$\{2.6, 3.5, 4.0, 4.8\}$	$\{9, 10, 10, 10\}$
128	64	5.1	$\{6.7, 5.3, 8.3, 9.6\}$	$\{8, 10, 10, 9\}$
128	128	24.5	$\{13.8, 17.4, 15.2, 20.4\}$	$\{10, 9, 10, 7\}$
256	16	0.4	$\{4.3, 2.6, 3.3, 4.7\}$	$\{10, 8, 10, 9\}$
256	32	1.8	$\{6.3, 8.2, 7.9, 8.8\}$	$\{8, 9, 10, 10\}$
256	64	9.2	$\{12.4, 10.1, 19.2, 16.3\}$	$\{10, 10, 10, 9\}$
256	128	39.9	$\{33.5, 36.3, 30.6, 35.6\}$	$\{10, 9, 8, 9\}$
512	16	0.5	$\{6.7, 6.3, 6.1, 7.5\}$	$\{10, 9, 10, 10\}$
512	32	2.4	$\{16.6, 19.6, 20.2, 17.5\}$	$\{9, 9, 10, 10\}$
512	64	13.6	$\{34.9, 23.5, 22.3, 41.0\}$	$\{10, 10, 8, 7\}$
512	128	68.0	$\{74.5, 68.1, 72.3, 62.8\}$	$\{10, 1, 9, 9\}$

scaling [12, 21] and projection pursuit [6, 9]. Most recent research on this topic (also referred to as manifold learning) is still inspired by the aim of multi-dimensional scaling; find a low-dimensional embedding of points such that their distances in the high-dimensional space are well represented. In contrast to Principal Component Analysis [15], one usually does not treat all distances equal. Rather, the idea is to preserve small distances well, while large distances are irrelevant, as long as they remain large; examples are Local Linear and (t-)Stochastic Neighbor Embedding [8, 19, 22]. Even that is typically not possible to achieve perfectly, and a trade-off between precision and recall arises [24]. Recent works are mostly spectral methods along this line.

Iterative data mining and machine learning.

There are two general frameworks for iterative data mining: FORSIED [3, 4] is based on modeling the belief state of the user as an evolving probability distribution in order to formalize subjective interestingness of patterns. This distribution is chosen as the Maximum Entropy distribution subject to the user beliefs as constraints, at that moment in time. Given a pattern syntax, one then aims to find the pattern that provides the most information, quantified as the 'subjective information content' of the pattern.

The other framework, which we here named CORAND [7, 13], is similar, but the evolving distribution does not necessarily have an explicit form. Instead, it relies on sampling, or put differently, on randomization of the data, given the user beliefs as constraints. Both these frameworks are *general* in the sense that it has been shown they can be applied in various data mining settings; local pattern mining, clustering, dimensionality reduction, etc.

The main difference is that in FORSIED, the background model is expressed analytically, while in CORAND it is defined implicitly. This leads to differences in how they are deployed and when they are effective. From a research and development perspective, randomization schemes are easier to propose, or at least they require little mathematical skills. Explicit models have the advantage that they often enable faster search of the best pattern, and the models may be more transparent. Also, randomization schemes are computationally demanding when many randomizations are required. Yet, in cases like the current paper, a single randomization suffices, and the approach scales very well. For both frameworks, it is ultimately the pattern syntax that determines their relative tractability.

Besides FORSIED and CORAND, many special-purpose methods have been developed for active learning, a form of iterative mining or learning, in diverse settings: classification, ranking, and more, as well as explicit models for user preferences. However, since these approaches are not targeted at data exploration, we do not review them here. Finally, several special-purpose methods have been developed for visual iterative data exploration in specific contexts, for example for itemset mining and subgroup discovery [1, 5, 23, 14], information retrieval [20], and network analysis [2].

Visually controllable data mining.

This work was motivated by and can be considered an instance of *visually controllable data mining* [17], where the objective is to implement advanced data analysis method so that they are understandable and efficiently controllable by the user. Our proposed method satisfies the properties of a visually controllable data mining method (see [17], Section II B): (VC1) the data and model space are presented visually, (VC2) there are intuitive visual interactions that allow the user to modify the model space, and (VC3) the method is fast enough to allow for visual interaction.

Information visualization and visual analytics.

Many new interactive visualization methods are presented yearly at the IEEE Conference on Visual Analytics Science and Technology (VAST). The focus in these communities is not on the use or development of advanced data mining or machine learning techniques, and more on human cognition and efficient use of displays, as well as efficient exploration via selection of data objects and features. Yet, the need to interact with the data mining community was already recognized long ago [11].

5. CONCLUSIONS

In order to improve the efficiency and efficacy of data exploration, there is a growing need for generic tools that integrate advanced visualization with data mining techniques to facilitate effective visual data analysis by human users. Our aim with this paper was to present a proof of concept for how this need can be addressed: a tool that initially presents the user with an 'interesting' projection of the data and then employs data randomization with constraints to allow users to flexibly express their interests or beliefs. These constraints expressed by the user are then taken into account by a projection-finding algorithm to compute a new 'interesting' projection, a process that can be iterated until the user runs out of time or finds that constraints explain everything the user needs to know about the data.

In our example, the user can associate two types of constraints on a chosen subset of data points: the appearance of the points in the particular projection or the fact that the points can be nearby also in other projections. We also tested the tool on two data sets, one controlled experiment on synthetic data and another on real census data. We found that the tool performs according to our expectations; it manages to find interesting projections. Yet, interestingness can be case specific and relies on the definition of an appropriate interestingness measure, here the L_1 norm was employed. More research into this choice is warranted. Nonetheless, we think this approach is useful in constructing new tools and methods for interactive visually controllable data mining in variety of settings.

In further work we intend to investigate the use of the FORSIED framework to also formalize an analytical background model [3, 4], as well as its use for computing the most informative data projections. Additionally, alternative pattern syntaxes (constraints) will be investigated.

Acknowledgements.

This work was supported by the European Union through the ERC Consolidator Grant FORSIED (project reference 615517), Academy of Finland (decision 288814), and Tekes (Revolution of Knowledge Work project).

6. REFERENCES

[1] M. Boley, M. Mampaey, B. Kang, P. Tokmakov, and S. Wrobel. One click mining—interactive local pattern discovery through implicit preference and performance learning. In *Proc. of KDD IDEA*, pages 27–35, 2013.

[2] D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *Proc. of CHI*, pages 167–176, 2011.

[3] T. De Bie. An information-theoretic framework for data mining. In *Proc. of KDD*, pages 564–572, 2011.

[4] T. De Bie. Subjective interestingness in exploratory data mining. In *Proc. of IDA*, pages 19–31, 2013.

[5] V. Dzyuba and M. van Leeuwen. Interactive discovery of interesting subgroup sets. In *Proc. of IDA*, pages 150–161, 2013.

[6] J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Tr. Comp.*, 100(23):881–890, 1974.

[7] S. Hanhijärvi, M. Ojala, N. Vuokko, K. Puolamäki, N. Tatti, and H. Mannila. Tell me something I don't know: Randomization strategies for iterative data mining. In *Proc. of KDD*, pages 379–388, 2009.

[8] G. E. Hinton and S. T. Roweis. Stochastic neighbor embedding. In *Proc. of NIPS*, pages 857–864, 2003.

[9] P. J. Huber. Projection pursuit. *Ann. Stat.*, 13(2):435–475, 1985.

[10] B. Kang, K. Puolamäki, J. Lijffijt, and T. De Bie. A tool for subjective and interactive visual data exploration. Under review.

[11] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, editors. *Mastering the Information Age: Solving Problems with Visual Analytics*. Eurographics Association, 2010.

[12] J. B. Kruskal. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29(2):115–129, 1964.

[13] J. Lijffijt, P. Papapetrou, and K. Puolamäki. A statistical significance testing approach to mining the most informative set of patterns. *DMKD*, 28(1):238–263, 2014.

[14] D. Paurat, R. Garnett, and T. Gärtner. Interactive exploration of larger pattern collections: A case study on a cocktail dataset. In *Proc. of KDD IDEA*, pages 98–106, 2014.

[15] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901.

[16] K. Puolamäki, B. Kang, J. Lijffijt, and T. De Bie. Interactive visual data exploration with subjective feedback. Under review.

[17] K. Puolamäki, P. Papapetrou, and J. Lijffijt. Visually controllable data mining methods. In *Proc. of ICDMW*, pages 409–417, 2010.

[18] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.

[19] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[20] T. Ruotsalo, G. Jacucci, P. Myllymäki, , and S. Kaski. Interactive intent modeling: Information discovery beyond search. *CACM*, 58(1):86–92, 2015.

[21] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

[22] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 9(Nov):2579–2605, 2008.

[23] M. van Leeuwen and L. Cardinaels. Viper — visual pattern explorer. In *Proc. of ECML–PKDD*, pages 333–336, 2015.

[24] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *JMLR*, 11(Feb):451–490, 2010.