Interactive Knowledge-Based Kernel PCA^{*}

Dino Oglic¹, Daniel Paurat¹, and Thomas Gärtner^{1,2}

¹ University of Bonn, Bonn, Germany

² Fraunhofer IAIS, Sankt Augustin, Germany

{dino.oglic, daniel.paurat, thomas.gaertner}@uni-bonn.de

Abstract. Data understanding is an iterative process in which domain experts combine their knowledge with the data at hand to explore and confirm hypotheses. One important set of tools for exploring hypotheses about data are visualizations. Often, however, traditional, unsupervised dimensionality reduction algorithms are used for visualization. These tools allow for interaction, i.e., exploring different visualizations, only by means of manipulating some technical parameters of the algorithm. Therefore, instead of being able to intuitively interact with the visualization, domain experts have to learn and argue about these technical parameters. In this paper we propose a knowledge-based kernel PCA approach that allows for intuitive interaction with data visualizations. Each embedding direction is given by a non-convex quadratic optimization problem over an ellipsoid and has a globally optimal solution in the kernel feature space. A solution can be found in polynomial time using the algorithm presented in this paper. To facilitate direct feedback, i.e., updating the whole embedding with a sufficiently high frame-rate during interaction, we reduce the computational complexity further by incremental up- and down-dating. Our empirical evaluation demonstrates the flexibility and utility of this approach.

Keywords: Interactive visualization, kernel methods, dimensionality reduction.

1 Introduction

We investigate a variant of kernel principal component analysis (PCA) which allows domain experts to directly interact with data visualizations and to add domain-knowledge and other constraints in an intuitive way. Data visualization is an important part of knowledge discovery and at the core of data understanding and exploration tasks (see, e.g., (author?) [22]). Its importance for data science has been recognized already by (author?) [24]. While knowledge discovery is inherently interactive and iterative, most data visualizations are inherently static (we survey several methods in Section 2). Switching methods and changing algorithmic parameters allow for some interaction with the visualization but this interaction is rather indirect and only feasible for machine learning experts rather than domain experts. As data science and its tools are, however, getting more and more widespread, the need arises for tools that allow domain experts to directly interact with the data.

^{*} Parts of this work have been presented in workshops [18, 19].

Our interactive and knowledge-based variant of kernel PCA (described in detail in Section 3) incorporates different forms of supervision either as soft- or hard constraints. In particular, it allows for the placement of ‘control points’, the addition of must-link and cannot-link constraints, as well as being able to incorporate known class labels. Here, the motivation behind constraints is that the domain experts can choose one low dimensional embedding from the many possible ones not by tuning parameters but by dragging chosen data points in the embedding or grouping them by similarity and class labels whereby all related data points automatically and smoothly adjust their location accordingly. Similar to kernel PCA, each embedding direction/dimension corresponds to a function in the underlying reproducing kernel Hilbert space. For each direction, we propose to find a function which (i) maximizes the variance along this direction, (ii) has unit norm, (iii) is as orthogonal to the other functions as possible, and (iv) adheres to the knowledge-based constraints as much as possible.

The optimization problem (analyzed in detail in Section 4) derived from the above formulation is to maximize a—typically non-convex—quadratic function over an ellipsoid subject to linear equality constraints. Unconstrained quadratic optimization problems over spheres have been investigated by (author?) [11] who generally suggested two approaches: transforming the problem to a quadratic and then linear eigenvalue problem or reducing the problem to solving a one-dimensional secular equation. While both approaches have cubic complexity and the first approach is more elegant, the second one is numerically much more stable. To solve the quadratic function over ellipsoid subject to linear constraints, we extend the approach of (author?) [11] to this setting.

In order to allow a direct interaction with the embedding, i.e., updating the whole embedding with a sufficiently high frame-rate, the cubic complexity of the just mentioned approaches is, however, not sufficient. To overcome this, we observe that in an interactive set up it is hardly ever the case that the optimization problem has to be solved from scratch. Instead, consecutive optimization problems will be strongly related and indeed we show (in Section 5) that consecutive solutions differ only in rank-one updates which allows for much more fluent and natural interaction.

Our experiments focus on demonstrating the flexibility and usability of our approach in an interactive knowledge discovery setting (in Section 6). We show first that small perturbations of the location of control points only lead to small perturbations of the embedding of all points. This directly implies that it is possible to smoothly change the embedding without sudden and unexpected jumps (large changes) of the visualization. We then show that by appropriate placement of control points, knowledge-based kernel PCA can mimic other embeddings. In particular, we consider the sum of two different kernels and observe that by placing a few control points, the 2D kernel PCA embedding of either of the original kernels can be reasonably well recovered. In addition, we investigate the amount of information retained in low-dimensional embeddings. For that we embed benchmark semi-supervised classification data sets in 2D space and compare the predictive performance of learning algorithms in this embedding with other approaches. Last but not least, we show it is possible to discover structures within the data that do not necessarily exhibit high correlation with variance and, thus, remain hidden in the plain kernel PCA embedding.

2 Related Work

Interaction with traditional machine learning algorithms is hardly intuitive and forces domain experts to reason about parameters of the algorithm instead of data. To overcome this problem several tools for data visualization were designed with the ultimate goal to facilitate the understanding of the underlying model and the interaction with model parameters. One such tool facilitates the understanding of PCA by interpreting the influence of the movement of a point in the projection space along the principal directions on the coordinates of the corresponding instance in the input space [14]. The interaction with the PCA is, however, limited to feature weighting which allows domain experts to reason on the importance of particular features in the analyzed data set. In contrast, a user study conducted by (author?) [1] shows domain experts prefer to interact directly with a visualization by placing a few control points in accordance with the current understanding of data. In general, user studies report benefits of the interactive over the static approach in data visualization (see, e.g., (author?) [4, 14]).

In this paper we extend our previous work on interactive visualization in which we proposed a variant of supervised PCA [19] and provided a tool for interactive data visualization InVis [18]. In contrast to supervised PCA [19] which allowed interaction with a visualization only through the explicit placement of control points, the proposed knowledge-based kernel PCA allows interaction through a variety of soft and/or hard knowledge-based constraints (see Section 3.1, 3.2 or 3.3). Moreover, we relate the successive optimization problems arising during the interaction with rank-one updates and reduce the interaction complexity from cubic to quadratic in the number of data instances. The InVis tool, on the other hand, enables interaction with a visualization using the least square projections (LSP). As argued in our workshop paper [19], the LSP algorithm is in general not a good choice for data visualization and the same can be said about any purely supervised learning algorithm (e.g. linear discriminant analysis [13]). For instance, consider training a linear regression on a sparse high dimensional dataset. If it is trained using very few instances, the weight vector will only be non-zero over the union of their non-zero attributes and all instances having different non-zero attributes will be mapped to the origin—a pretty cluttered visualization.

From the interactive visualization perspective, the most related to our work are techniques developed by (author?) [8] and (author?) [16]. The proposed techniques allow placement of control points in a projection space, but the placement induces a different form of interaction. Namely, movement of points is interpreted as a similarity feedback (similar to must and cannot link constraints) and as such incorporated into the model parameters. Our approach, on the other hand, focuses on the explicit placement of control points enabling structural exploration of the data by observing how the embedding reacts to a placement of a selected control point. Both approaches [8, 16] perform interaction by incorporating expert feedback into the probabilistic PCA, multi-dimensional scaling and generative topographic mapping. The means to incorporate the similarity feedback into the named unsupervised algorithms are, however, limited to feature weighting and heuristic covariance estimates. Moreover, probabilistic PCA and multi-dimensional scaling approaches are highly sensitive to outliers and produce visualizations with huge number of overlapping points for such data sets.

From the methodological perspective, our method is a spectral method for semi-supervised learning and closely related to the semi-supervised kernel PCA [25]. This method can be described as a relaxation of T-SVM and/or a generalization of kernel PCA. Three such generalizations are proposed by (author?) [25] and the closest to our approach is the least square kernel PCA which can be viewed as choosing a projection with the best L_2 -fit over the labelled data among the projections having a constant variance.

Methods for dimensionality reduction are also related to data visualization, and in many cases used primarily for visualization purposes. There are several well known methods for dimensionality reduction, but the majority of them is unsupervised and unable to incorporate the gained knowledge into a lower dimensional data representation. Some of the well known spectral methods for dimensionality reduction are principal component analysis [15], metric multidimensional scaling [6], isomap [23], maximum variance unfolding [26] and locally linear embedding [20]. Here we note that these methods can be also viewed as instances of kernel PCA with a suitably defined kernel matrix [12].

3 Knowledge-Based Kernel PCA

In this section we present a variant of semi-supervised kernel PCA which incorporates various domain-knowledge constraints while still maximizing the variance of the data ‘along’ the set of unit norm functions defining the embedding. Our formulation can take into account a variety of hard and/or soft constraints, allowing flexible placement of control points in an embedding space, the addition of must-link and cannot-link constraints, as well as known class labels. The goal of the proposed algorithm is to allow domain experts to interact with a low-dimensional embedding and choose from the many possible ones not by tuning parameters but by dragging or grouping the chosen data points in the embedding whereby all related data points automatically and smoothly adjust their location accordingly. To make the visualization more flexible with respect to orthogonality of maximum-variance directions we replace the usual hard orthogonality constraint with a soft-orthogonality term in the objective function.

Let $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ be a sample from an instance space $\mathcal{X}$, $\mathcal{H}$ a reproducing kernel Hilbert space with kernel $k(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ and $\mathcal{H}_X = \text{span}\{k(\mathbf{x}_i, \cdot) | \mathbf{x}_i \in X\}$. We iteratively find the constant $\mathcal{H}_X$ -norm maximum variance directions $f_1, \dots, f_d$ by solving the following optimization problem

$$\begin{aligned} f_s &= \underset{f \in \mathcal{H}}{\operatorname{argmax}} \quad \frac{1}{n} \sum_{i=1}^n (f(\mathbf{x}_i) - \langle f, \mu \rangle)^2 + \rho \Omega(f, s) - \nu \sum_{s'=1}^{s-1} \langle f_{s'}, f \rangle^2 \\ \text{subject to} \quad & \|f\|_{\mathcal{H}_X} = r, \\ & \Upsilon(f, \mathbf{y}_s) = 0, \end{aligned} \tag{1}$$

where Ω is a soft and Υ is a hard constraint term, $r \in \mathbb{R}^+$, $\mu = \frac{1}{n} \sum_{i=1}^n k(\mathbf{x}_i, \cdot)$ and $\mathbf{y}_s \in \mathbb{R}^m$ ($m < n$) is chosen interactively. Here we note that Υ is a linear operator over a direction f evaluated at $\mathbf{x} \in X$. Additionally, the term can be used to express a hard orthogonality over the computed directions.

First, let us see that the weak representer theorem [7, 21] holds. As an optimizer $f_s \in \mathcal{H}$ we can write $f_s = u_s + v_s$ with $u_s \perp v_s$ and $u_s \in \mathcal{H}_X$. For the computation of the first extreme variance direction, f_1 , there is no soft orthogonality term in the optimization objective. Plugging the substitution into Eq. (1) we conclude, provided the theorem holds for the Ω term, that the optimization objective is independent of v_s , and the weak representer theorem holds in this case. For the computation of the s -th variance direction f_s ($s > 1$), we additionally have orthogonality terms $\langle f_s, f_{s'} \rangle = \langle u_s + v_s, f_{s'} \rangle = \langle u_s, f_{s'} \rangle$ ($s' < s$) which are also independent of v_s . The hard constraint term Υ is also independent of v_s as it holds that $f_s(x) = u_s(x)$ for all $x \in X$. Therefore, the weak representer theorem holds for problem (1) and we can express an optimizer as $f_s = \sum_i \alpha_{si} k(\mathbf{x}_i, \cdot)$ with $\alpha_{si} \in \mathbb{R}$.

Now, using the representer theorem we can rewrite terms from problem (1) as:

$$\begin{aligned} \langle f, \mu \rangle &= \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \alpha_i k(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{n} \mathbf{e}^T K \alpha, \\ \sum_{s'=1}^{s-1} \langle f, f_{s'} \rangle^2 &= \alpha^T K \left(\sum_{s'=1}^{s-1} \alpha_{s'} \alpha_{s'}^T \right) K \alpha, \end{aligned}$$

with $\mathbf{e}$ denoting the vector of ones. If we ignore the hard constraint term (in Section 4.2 we show we can incorporate it into the objective) we can write problem (1) as:

$$\begin{aligned} \alpha_s &= \underset{\alpha \in \mathbb{R}^n}{\operatorname{argmax}} \quad \alpha^T K W K \alpha + \rho \Omega(\alpha, s) \\ \text{subject to} \quad & \alpha^T K \alpha = r^2, \end{aligned} \quad (2)$$

with $H_n = \mathbf{I}_n - \frac{1}{n} \mathbf{e} \mathbf{e}^T$ and $W = \frac{1}{n} H_n - \nu \sum_{s'=1}^{s-1} \alpha_{s'} \alpha_{s'}^T$.

In the following sections we introduce several, quadratic and linear, knowledge-based constraints satisfying the weak representer theorem and enabling different forms of interaction with a low-dimensional embedding.

3.1 The Placement of Control Points

The most natural form of interaction with a low-dimensional embedding is the movement of control points across the projection space [1]. It is an exploratory form of interaction enabling domain experts to explore the structure of the data by observing how the embedding reacts to a movement of a selected control point. The placement of control points can be incorporated into the kernel PCA as a ‘soft’ constraint with

$$\Omega(f, s) = -\frac{1}{m} \sum_{i=1}^m \|f(\mathbf{x}_i) - y_{si}\|^2,$$

where y_{si} denotes the coordinate ‘along’ the projection axis f_s of an example $\mathbf{x}_i$. That the weak representer theorem holds for problem (1), including this Ω term, follows along the same lines as above and we are able to express Ω as

$$\Omega(\alpha, s) = -\frac{1}{m} (\alpha^T K_{[:,m]} K_{[:,m]} \alpha - 2 \mathbf{y}_s^T K_{[:,m]} \alpha), \quad (3)$$

with $K_{[:,m]}$ denoting rows in the kernel matrix K corresponding to control points.

The alternative way is to treat the placements as hard constraints [19] and incorporate them into $\mathcal{T}$ term which can be written as

$$\mathcal{T}(f, \mathbf{y}_s) = K_{[:m, :n]} \boldsymbol{\alpha} - \mathbf{y}_s = 0. \quad (4)$$

Note that the soft constraint Ω is allowing some displacement which can lead to better visualizations if noise is to be expected in the positions of the control points.

3.2 Must-Link and Cannot-Link Constraints

Domain knowledge can also be expressed in terms of similarity between points and the knowledge-based term Ω can, for instance, be defined by pairing points which should or should not be placed close to each other. Squared distances between projections of paired points are then minimized for must-link pairs and maximized for cannot-link pairs, i.e.

$$\Omega(f, s) = -\frac{1}{|\mathcal{C}|} \sum_{(i,l) \in \mathcal{C}} y_{il} (f(\mathbf{x}_i) - f(\mathbf{x}_l))^2,$$

where $y_{il} = +1$ for a must-link and $y_{il} = -1$ for a cannot-link constraint, and $\mathcal{C}$ denotes the set of constraints. Analogously to the previous case, the weak representer theorem holds and the constraint can be written as

$$\Omega(\boldsymbol{\alpha}, s) = -\frac{1}{|\mathcal{C}|} \boldsymbol{\alpha}^T K L K \boldsymbol{\alpha}. \quad (5)$$

Here $\boldsymbol{\Delta}_{il} = \mathbf{e}_i - \mathbf{e}_l$ and $L = \sum_{(i,l) \in \mathcal{C}} y_{il} \boldsymbol{\Delta}_{il} \boldsymbol{\Delta}_{il}^T$ is a Laplacian matrix of the graph weighted with y_{il} .

3.3 Classification Constraints

In this case domain-knowledge is incorporated by providing positive and negative class labels for a small number of instances and the ‘soft’ knowledge-based term Ω can be written as

$$\Omega(f, s) = \langle f, \mu_{\pm} \rangle, \text{ with } \mu_{\pm} = \frac{1}{m} \sum_{i=1}^m y_i k(\mathbf{x}_i, \cdot).$$

Similar to previous cases, it can be checked that the weak representer theorem holds and the Ω term can be written as

$$\Omega(\boldsymbol{\alpha}, s) = \frac{1}{m} \mathbf{y}^T K_{[:m, :n]} \boldsymbol{\alpha}. \quad (6)$$

4 Optimization Problem

As stated in Section 3, it is possible to combine different knowledge-based constraints and for any such combination we are interested in the resulting optimization problem.

Each of the ‘soft’ knowledge-based constraints is either quadratic or linear and contributes with a quadratic or linear term to the optimization objective. Therefore, to compute our embedding, for any combination of ‘soft’ knowledge-based constraints, we have to solve the following optimization problem:

$$\operatorname{argmax}_{\mathbf{x} \in \mathbb{R}^n} \mathbf{x}^T W \mathbf{x} - 2\mathbf{b}^T \mathbf{x} \quad \text{s. t. } \mathbf{x}^T K \mathbf{x} = r^2. \quad (7)$$

where $W \in \mathbb{R}^{n \times n}$ is a symmetric and $K \in \mathbb{R}^{n \times n}$ is a kernel matrix. Problem (7) is non-convex as it is a quadratic defined over an ellipsoid. If we also want to force the hard orthogonality or/and hard placement of control points we need to add an additional linear constraint

$$L\mathbf{x} = \mathbf{y}, \quad (8)$$

where $L \in \mathbb{R}^{m \times n}$ is a rectangular constraint matrix and $\mathbf{y}$ is a coordinate placement along one of the projection axis or the zero vector (in hard orthogonality case).

In Section 4.1 we describe how to find the global optimizer for the non-convex problem (7) in a closed form.

4.1 Quadratic over a Hypersphere

We first transform problem (7) to optimize over a hypersphere instead of hyperellipsoid. To achieve this we decompose the positive definite matrix K and introduce a substitution $\mathbf{v} = K^{\frac{1}{2}} \mathbf{x}$. In the new optimization problem the symmetric matrix W is replaced with the symmetric matrix $C = K^{-\frac{1}{2}} W K^{-\frac{1}{2}}$ and the vector $\mathbf{b}$ with the vector $\mathbf{d} = K^{-\frac{1}{2}} \mathbf{b}$. Hence, after the transformation we are optimizing a quadratic over the hypersphere,

$$\operatorname{argmax}_{\mathbf{v} \in \mathbb{R}^n} \mathbf{v}^T C \mathbf{v} - 2\mathbf{d}^T \mathbf{v} \quad \text{s. t. } \mathbf{v}^T \mathbf{v} = r^2. \quad (9)$$

To solve the problem we form the Lagrange function

$$\mathcal{L}(\mathbf{v}, \lambda) = \mathbf{v}^T C \mathbf{v} - 2\mathbf{d}^T \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - r^2), \quad (10)$$

and set its derivatives to zero, i.e.

$$C\mathbf{v} = \mathbf{d} + \lambda\mathbf{v}, \quad \mathbf{v}^T \mathbf{v} = r^2. \quad (11)$$

As this is a non-convex problem a solution of the system in Eq. (11) is only a local optimum for problem (9). The following lemma, however, gives a criterion for distinguishing the global optimum of problem (9) from the solution set of the system in Eq. (11). Alternative and slightly more complex proofs for the same claim are given by (author?) [10] and (author?) [9]. Let us now denote the optimization objective of Eq. (9) with $\chi(\mathbf{v})$.

Lemma 1. *The maximum of the function $\chi(\mathbf{v})$ is attained at the tuple $(\mathbf{v}, \lambda)$ satisfying the stationary constraints (11) with the largest value of λ .*

Proof. Let $(\mathbf{v}_1, \lambda_1)$ and $(\mathbf{v}_2, \lambda_2)$ be two tuples satisfying the stationary constraints (11) with $\lambda_1 \geq \lambda_2$. Plugging the tuples into the first stationary constraint we obtain

$$C\mathbf{v}_1 = \lambda_1\mathbf{v}_1 + \mathbf{d}, \quad (12)$$

$$C\mathbf{v}_2 = \lambda_2\mathbf{v}_2 + \mathbf{d}. \quad (13)$$

Subtracting (13) from (12) we have

$$C\mathbf{v}_1 - C\mathbf{v}_2 = \lambda_1\mathbf{v}_1 - \lambda_2\mathbf{v}_2. \quad (14)$$

Multiplying (14) first with $\mathbf{v}_1^T$ and then with $\mathbf{v}_2^T$ and adding the resulting two equations (having in mind that the matrix C is symmetric) we deduce

$$\mathbf{v}_1^T C\mathbf{v}_1 - \mathbf{v}_2^T C\mathbf{v}_2 = (\lambda_1 - \lambda_2)(r^2 + \mathbf{v}_1^T \mathbf{v}_2). \quad (15)$$

On the other hand, using the Cauchy-Schwarz inequality and (11) we deduce

$$\mathbf{v}_1^T \mathbf{v}_2 \leq \|\mathbf{v}_1\| \|\mathbf{v}_2\| = r^2. \quad (16)$$

Now, combining the results obtained in (15) and (16) with the initial assumption $\lambda_1 \geq \lambda_2$ we deduce

$$\mathbf{v}_1^T C\mathbf{v}_1 - \mathbf{v}_2^T C\mathbf{v}_2 \leq 2r^2(\lambda_1 - \lambda_2). \quad (17)$$

Finally, subtracting the optimization objectives for the two tuples and using (12) and (13) multiplied by $\mathbf{v}_1^T$ and $\mathbf{v}_2^T$, respectively, we prove

$$\chi(\mathbf{v}_1) - \chi(\mathbf{v}_2) = 2r^2(\lambda_1 - \lambda_2) - (\mathbf{v}_1^T C\mathbf{v}_1 - \mathbf{v}_2^T C\mathbf{v}_2) \geq 0,$$

where the last inequality follows from (17). $\square$

Hence, instead of the original optimization problem (9) we can solve the system with two stationary equations (11) with maximal λ . (author?) [11] propose two methods for solving such problems. In the first approach, the problem is reduced to a quadratic eigenvalue problem and afterwards transformed into a linear eigenvalue problem. In the second approach the problem is reduced to solving a one-dimensional secular equation. The first approach is more elegant, as it allows us to compute the solution in a closed form. Namely, the solution to the problem (9) is given by [11]

$$\mathbf{v}^* = (C - \lambda_{max}\mathbf{I})^{-1}\mathbf{d},$$

where λ_{max} is the largest real eigenvalue of

$$\begin{bmatrix} C & -\mathbf{I} \\ -\frac{1}{r^2}\mathbf{d}\mathbf{d}^T & C \end{bmatrix} \begin{bmatrix} \gamma \\ \eta \end{bmatrix} = \lambda \begin{bmatrix} \gamma \\ \eta \end{bmatrix}.$$

Despite its elegance, the approach requires us to decompose a non-symmetric block matrix of dimension $2n$ and this is not a numerically stable task for every such matrix. Furthermore, the computed solution $\mathbf{v}^*$ highly depends on the precision up to which

the optimal λ is computed and for an imprecise value the solution might not be on the hypersphere at all (for a detailed study refer to (author?) [11]).

For this reason, we rely on the secular approach in the computation of the optimal solution. In Section 5 we deal with an efficient algorithm for the computation of the parameter to a machine precision and here we describe how to derive the secular equation required to compute the optimal λ . In the first step the stationary constraint (11) is simplified by decomposing the symmetric matrix $C = P\Delta P^T$, i.e. $P\Delta P^T \mathbf{v} = \mathbf{d} + \lambda \mathbf{v}$. Then, the resulting equation is multiplied with the orthogonal matrix P^T from the left and transformed into $\Delta \mathbf{t} = \hat{\mathbf{d}} + \lambda \mathbf{t}$, with $\hat{\mathbf{d}} = P^T \mathbf{d}$ and $\mathbf{t} = P^T \mathbf{v}$. From the last equation we compute

$$t_i(\lambda) = \hat{d}_i / (\Delta_{ii} - \lambda) \quad (i = 1, 2, \dots, n),$$

and substitute the computed $\mathbf{t}$ -vector into the second stationary constraint to form the secular equation

$$g(\lambda) = \sum_i t_i^2(\lambda) - r^2 = 0. \quad (18)$$

The optimal value of parameter λ is the largest root of the non-linear secular equation and the optimal solution to problem (9) is given by $\mathbf{v}^* = P \cdot \mathbf{t}(\lambda_{max})$. Moreover, the interval at which the root lies is known [11]. Namely, it must hold $\lambda_{max} \geq \Delta_{11}$, where Δ_{11} is the largest eigenvalue of the symmetric matrix C .

The complexity of both approaches (secular and eigenvalue) for a d -dimensional embedding is $O(dn^3)$, where n is the number of data instances. The cubic term arises from the eigendecompositions required to compute the solutions to problem (9) for each of the d variance direction.

In Section 4.2 we show how to transform the optimization problem (1) with additional hard constraints to optimize only problem (9).

4.2 Eliminating a Linear Constraint

For the optimization problems involving hard orthogonality or hard placement of control points we have an additional linear term (8) in the optimization problem (7). If the linear term is of rank $m < n$, we can eliminate it and transform the problem to optimize a quadratic over an $(n - m)$ -dimensional hypersphere. The linear constraint term is first transformed from a constraint over a hyperellipsoid to a constraint over a hypersphere by replacing the matrix L (the linear constraint in Eq. (8)) with the matrix $LK^{-\frac{1}{2}}$. In the remainder of the section, we will denote the transformed constraint matrix with L .

In order to eliminate the linear constraint we do a QR factorization of the matrix $L^T = QR$, where $Q \in \mathbb{R}^{n \times n}$ is an orthogonal and $R = \begin{bmatrix} \bar{R} \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^{n \times m}$ is an upper-diagonal matrix. Substituting

$$Q^T \mathbf{v} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}, \text{ with } \mathbf{z}_1 \in \mathbb{R}^m \text{ and } \mathbf{z}_2 \in \mathbb{R}^{n-m},$$

linear and sphere constraints are transformed into

$$\begin{aligned} \mathbf{y} = L\mathbf{v} = R^T(Q^T\mathbf{v}) = \overline{R}^T\mathbf{z}_1 &\implies \mathbf{z}_1 = (\overline{R}^T)^{-1}\mathbf{y}, \\ r^2 = \mathbf{v}^T\mathbf{v} = \mathbf{z}_1^T\mathbf{z}_1 + \mathbf{z}_2^T\mathbf{z}_2 &\implies \mathbf{z}_2^T\mathbf{z}_2 = r^2 - \mathbf{z}_1^T\mathbf{z}_1 = \hat{r}^2. \end{aligned}$$

As $\mathbf{z}_1$ is a constant vector we can rewrite the objective in Eq. (9) as a quadratic over an $(n - m)$ -dimensional sphere. Namely, the quadratic term can be rewritten as

$$\begin{aligned} \mathbf{v}^T C \mathbf{v} &= \mathbf{v}^T Q Q^T C Q Q^T \mathbf{v} = \mathbf{z}_1^T F \mathbf{z}_1 + 2\mathbf{z}_2^T G \mathbf{z}_1 + \mathbf{z}_2^T H \mathbf{z}_2, \text{ where} \\ Q^T C Q &= \begin{bmatrix} F & G^T \\ G & H \end{bmatrix} \text{ with } F \in \mathbb{R}^{m \times m}, G \in \mathbb{R}^{(n-m) \times m} \text{ and } H \in \mathbb{R}^{(n-m) \times (n-m)}. \end{aligned}$$

On the other hand, the linear term is transformed into

$$\mathbf{d}^T \mathbf{v} = \mathbf{d}^T Q Q^T \mathbf{v} = \mathbf{f}_1^T \mathbf{z}_1 + \mathbf{f}_2^T \mathbf{z}_2,$$

where $\mathbf{f}_1 \in \mathbb{R}^m$ and $\mathbf{f}_2 \in \mathbb{R}^{n-m}$ are blocks in the vector $Q^T \mathbf{d}$. Denoting with $\hat{\mathbf{d}} = \mathbf{f}_2 - G\mathbf{z}_1$ and $\hat{\mathbf{z}} = \mathbf{z}_2$ we obtain

$$\operatorname{argmax}_{\hat{\mathbf{z}} \in \mathbb{R}^{n-m}} \hat{\mathbf{z}}^T H \hat{\mathbf{z}} - 2\hat{\mathbf{d}}^T \hat{\mathbf{z}} \quad \text{s. t. } \hat{\mathbf{z}}^T \hat{\mathbf{z}} = \hat{r}^2,$$

for which a closed form solution was given in Section 4.1.

5 Numerically Efficient Interaction

To shape an embedding interactively with the help of knowledge-based constraints it is required to solve the optimization problem (1) at each interaction step. In Section 4 we have described how to solve the arising optimization problem with complexity $O(dn^3)$, where n is the number of instances and d is the number of variance directions. In this section, we show how the proposed algorithm can be improved to enable a user interaction in $O(d^2n^2)$ time. To achieve this, we express the interaction in the form of rank-one updates of the original problem and review a linear time algorithm for solving secular equations arising in the process.

5.1 Efficient Formulation of the Interaction

Let us assume, without loss of generality, the algorithm is working with the classification constraints and the soft orthogonality. The optimization problem for this setting can be expressed in the form (9). Furthermore, assume a user, interested in a d -dimensional embedding, has provided labels $\mathbf{y}_k$ at an interaction step k . To compute embeddings interactively the algorithm needs to solve problem (9) for different interaction steps and for all d directions. We denote with C the symmetric matrix defining the quadratic term in the problem arising in a step k for a direction s . The variance term, independent of the provided labels and denoted with $\overline{C}$, can be decomposed prior to any interaction. The decomposition has complexity $O(n^3)$, but it is a one time cost paid prior to interaction steps. The linear term for a direction s at the step k is a function of a block of the

kernel matrix and the label vector $\mathbf{y}_k$ and we denote it with $\mathbf{a}_k$. Now, the first stationary constraint of the problem (9) for a direction $s + 1$ at the step k can be written as

$$C' \mathbf{x} = \lambda \mathbf{x} + \mathbf{a}_k,$$

where the symmetric matrix C' is a rank-one update of the previous direction's quadratic term C for which the eigendecomposition is already computed. Reusing the decomposition, $C = U \Delta U^T$, we can rewrite the last equation as

$$C' = C - \mu \boldsymbol{\alpha}_s \boldsymbol{\alpha}_s^T = U (\Delta - \mu \mathbf{z} \mathbf{z}^T) U^T,$$

where $\boldsymbol{\alpha}_s$ denotes the s -th direction vector and $\mathbf{z} = U^T \boldsymbol{\alpha}_s$. Let us denote the rank-one update to the diagonal matrix as

$$\Theta = \Delta - \mu \mathbf{z} \mathbf{z}^T \quad (19)$$

The complexity of a full eigendecomposition (see, e.g., (author?) [2, 3]) of the matrix Θ is $O(n^2)$ and it is computed by solving n secular equations (see Section 5.2), one for each of the eigenvalues of the matrix Θ . Rewriting the last equation using the substitution we get

$$U \Theta U^T \mathbf{x} = \lambda \mathbf{x} + \mathbf{a}_k \implies \Theta \bar{\mathbf{x}} = \lambda \bar{\mathbf{x}} + \bar{\mathbf{a}}, \text{ with } \bar{\mathbf{x}} = U^T \mathbf{x} \text{ and } \bar{\mathbf{a}} = U^T \mathbf{a}_k.$$

Now, using the decomposition $\Theta = V \Delta' V^T$ we transform the last problem into

$$\Delta' \mathbf{t} = \lambda \mathbf{t} + \mathbf{f}, \text{ with } \mathbf{t} = V^T \bar{\mathbf{x}} \text{ and } \mathbf{f} = V^T \bar{\mathbf{a}}. \quad (20)$$

The second stationary constraint combined with Eq. (20) yields secular equation (18).

Thus, the $(s + 1)$ -th direction vector $\mathbf{x}_{s+1}$ is computed as

$$\mathbf{x}_{s+1} = \bar{U} \left(\prod_{i=1}^s V_i \right) \cdot \mathbf{t}(\lambda_{max}), \text{ with } t_i(\lambda) = \frac{(\bar{U} \prod_{i=1}^s V_i)^T \mathbf{a}_k}{\delta_i^{(s+1)} - \lambda} \quad (i = \overline{1, n}).$$

Note that $\bar{U}$ is the eigenvector matrix for the variance term $\bar{C}$ and $\delta_i^{(s+1)}$ is an eigenvalue of the quadratic term matrix for the direction $s + 1$ (see problems (1) and (9)).

Hence, to compute the directions at the interaction step k we need to perform $O(d^2)$ matrix vector multiplications, each incurring a quadratic cost, together with d quadratic time decompositions of Θ matrices. What remains to compute the data projection is a multiplication of direction vectors with the kernel matrix which is again of quadratic complexity. Therefore, the overall complexity of an interaction step is $O(d^2 n^2)$. In a similar fashion, it is possible to show the quadratic complexity of an interaction for other knowledge-based constraints.

5.2 Efficient and Stable Secular Solver

In this section, we review the state of the art in computation of the root of a secular equation in a given interval (δ_i, δ_{i+1}) , with $\delta_1 \leq \delta_2 \leq \dots \delta_{n-1} \leq \delta_n$. In particular, we are interested in finding a root of Eq. (18) and the equation [3]

$$\bar{g}(\lambda) = 1 + \mu \sum_{i=1}^n \frac{z_i^2}{\delta_i - \lambda},$$

whose roots are the eigenvalues of the perturbed matrix (19). In the discussion to follow we focus on finding the roots of $\bar{g}(\lambda)$ and digress to the secular function (18) only when the arguments are not eligible for it.

It is possible to compute a root of a secular equation with a linear number of flops [17]. An obvious choice for the root finder is a Newton method and, yet, it is not well suited for the problem. The tangent at certain points in the interval of interest crosses the x -axis outside this interval leading to incorrect solution or division by zero as the solution can converge to one of the poles δ_i . An efficient root finder, then, must overcome this issue and converge very quickly. The main idea behind the efficient root finder is to approximate a secular equation with a quadratic surrogate and find a zero of this surrogate. We go step-by-step through the procedure. First let us assume $\mu = 1$ and split the non-constant $\bar{g}$ terms into two functions

$$\psi_1(\lambda) = \sum_{k=1}^i \frac{z_k^2}{\delta_k - \lambda} \text{ and } \psi_2(\lambda) = \sum_{k=i+1}^n \frac{z_k^2}{\delta_k - \lambda}.$$

Then, each function ψ_k is approximated by a surrogate

$$h_k(x) = a_k + b_k/(\delta_{k-1+i} - x),$$

with constants computed such that at all candidate solutions x_j it holds [17]:

$$h_k(x_j) = \psi_k(x_j) \text{ and } h'_k(x_j) = \psi'_k(x_j).$$

Now, combining the surrogates for the particular terms we obtain the surrogate for the secular function $\bar{g}$,

$$\bar{h}(x) = c_3 + c_1/(\delta_i - x) + c_2/(\delta_{i+1} - x), \quad (21)$$

whose root is the next secular solution candidate. **(author?)** [3] proved the surrogate root finder converges to the desired root.

In contrast to the secular function $\bar{g}(\lambda)$, the secular equation (18) has only one surrogate [11]:

$$h(x) = p/(q - x)^2, h(x_j) = g(x_j) \text{ and } h'(x_j) = g'(x_j).$$

After computing the coefficients p and q we get the iteration step $\eta = x_{j+1} - x_j$ as

$$\eta = 2 \frac{g(x_j) + r^2}{g'(x_j)} \left(1 - \frac{\sqrt{g(x_j) + r^2}}{r} \right).$$

For the initial solution $\delta_1 < x_0 < \lambda^*$ the convergence is monotonic [3], i.e. $\lambda^* > x_{j+1} > x_j$, where λ^* is the desired root of the secular equation.

Finally, we note that different roots of secular Eq. (18) belong to distinct closed intervals and can be, therefore, computed in parallel. This enables an efficient GPU implementation of the secular solver resulting in a significant speed-up to the presented rank-one update algorithm. Consequently, with a GPU implementation of the secular solver it is possible to increase the interaction frame rate and improve scalability.

6 Experiments

The best, and ultimately only true, way to evaluate an algorithm for interactive data visualization and exploration is via a study with real domain experts that are using a tool implementing that algorithm. In the absence of the study we performed a number of *in silico* experiments which aim at illustrating the utility and sensibility of our approach. In particular, we show: (i) a rendered embedding is robust under small changes in the placement of control points; (ii) the approach is flexible in choosing a low-dimensional embedding from the many possible ones; (iii) ‘sufficient’ amount of information is retained in a visualization; (iv) it is possible to detect structures that do not necessarily exhibit high correlation with variance and which are, therefore, obscured in a regular kernel PCA embedding. We study the properties (i) – (iii) on benchmark data sets for semi-supervised learning [5] and generate an artificial data set to show the property (iv). In the experiments we use different kernels: Gaussian with a bandwidth equal to the median of pairwise distances between the instances, inverse Laplacian defined by the ϵ -neighbourhood graph, linear and polynomial kernel of degree three. All the reported results are averaged over ten runs.

Fig. 1. Distortion of an embedding over the perturbation of a control point

How stable is our approach? In exploratory data visualization it should be possible to smoothly change the embedding by moving control point throughout the projection space. In other words, small perturbations of a control point should result in small perturbations of the overall embedding. We empirically verify the stability of the proposed method by moving a control point randomly throughout the projection space. We express the displacement of a control point as a fraction of the median of pairwise distances within the embedding. The distortion or the difference between the two embeddings is measured by the average displacement of a point between them and this value is scaled by the median pairwise distance between the points in the kernel PCA embedding. In Figure 1 we show the distortion as the perturbation increases across five benchmark data sets [5]. Results clearly indicate that the proposed method provides stable visual exploration of data.

How flexible is our approach? It is possible to generate different embeddings of the same dataset with kernel PCA using different kernels. To show the flexibility of the proposed method we set up an experiment with a sum of different kernels and show that the proposed method can choose the PCA embedding corresponding to a kernel by re-arranging control points accordingly. In particular, we combine a Gaussian and the inverse Laplacian kernel that produce geometrically very different PCA embeddings of the considered datasets. We again measure the distortion between the two embeddings.

Fig. 2. Distortion between the target and the current embedding over the number of re-arranged control points. Results show, we can recover a target embedding with a small number of re-arranged control points.

The empirical results indicate that it is possible to recover the embeddings corresponding to PCA projection of each used kernel. Figure 2 shows the distortion between the current and the target embedding as the number of selected control points increases.

How informative is our approach? A satisfactory embedding should be able to retain a fair amount of information from the input space. To measure the amount of information the proposed method retains we simulate a classification task. We use the semi-supervised learning benchmark data set in which the small amount of labels is given to us. In building a classifier we use the classification constraints with a hard orthogonality and on top of it we apply the 1-NN classifier. We compare our method using only 1, 2 and 3 dimensional projections against the state-of-the-art unsupervised dimensionality reduction techniques with many more dimensions. In particular, dimensionality reduction algorithms use 38 dimensions for g241c, 4 dimensions for Digit1, 9 dimensions for USPS, 8 dimensions for BCI and 3 dimensions for COIL dataset [5]. The results indicate our algorithm is able to retain a satisfactory amount of information over the first 3 principal directions. We used Gaussian and inverse Laplacian kernel to compute the embeddings (see Figure 3).

Can we discover structures hidden by the plain PCA? We have created a 3D artificial data set to demonstrate the proposed approach is able to discover structures in data that do not exhibit high correlation with variance. Such structures remain hidden in the PCA projection and the proposed method is capable of detecting them by the appropriate placement of control points. In particular, we sample 3 plates of points from a 2D Gaussian distribution and embed these plates into the 3D space such that the z -axis coordinate for each plate is obtained by sampling from the normal distributions

Fig. 3. A classification error comparison between the proposed method and unsupervised dimensionality algorithms with many more dimensions [5]. Supervision was done with either 10 (greyed out) or 100 labels. We point out that the experiment was conducted to show the amount of information retained by the visualization and not to compete these methods.

Fig. 4. Discovering cluster structures which are hidden in the kernel PCA embedding

with means at 1, 0 and -1 . We choose the variance for the z -axis sampling such that the cluster-plates barely touch each other. For the sample generated in this way the within-clusters variance is higher than the between-cluster variance and the cluster structure remains hidden in the kernel PCA projection (see the first picture in Figure 4). Moving the two most distant points of each cluster apart (a total of 6 displacements) we discover the cluster structure obscured by the plain kernel PCA (the last picture in Figure 4).

Scalability. As stated in Section 5, it is possible to implement the interaction using rank-one updates with $O(n^2)$ complexity. Our experiments indicate that it is possible to interact with a visualization with multiple frames per second for datasets with ≈ 1000 instances. The frame rate and scalability can be significantly improved with an efficient GPU implementation of the secular solver (see Section 5.2). Moreover, it is possible to use a kernel expansion over a subset of instances while searching for a suitable placement of a control point and upon such placement the embedding can be computed with a full kernel expansion.

7 Conclusion

We proposed a novel, knowledge-based variant of kernel PCA with the aim of making visual data exploration a more interactive endeavour, in which users can intuitively modify the embedding by placing ‘control points’, adding must-link and cannot-link constraints, and supplying known class labels. We showed that maximizing the variance of unit norm functions that are as orthogonal as possible and that adhere as much as possible to the user supplied knowledge can be formulated as a non-convex quadratic optimization problem over an ellipsoid with linear constraints. We gave an algorithm for computing the resulting embedding in cubic time and argued that the more typical situation in interactive data analysis corresponds, however, to a low-rank update of the previous solution. We then derived an algorithm for computing these low-rank updates in quadratic time.

Acknowledgement. Part of this work was supported by the German Science Foundation (DFG) under the reference number ‘GA 1615/1-2’.

References

- [1] Andrews, C., Endert, A., North, C.: Space to Think: Large High-resolution Displays for Sensemaking. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI), pp. 55–64. ACM (2010)
- [2] Arbenz, P.: Lecture Notes on Solving Large Scale Eigenvalue Problems, pp. 77–93. ETH Zürich (2012)

- [3] Bunch, J.R., Nielsen, C.P., Sorensen, D.: Rank-one modification of the symmetric eigenproblem. *Numerische Mathematik* 31 (1978)
- [4] Callahan, E., Koenemann, J.: A comparative usability evaluation of user interfaces for on-line product catalog. In: *Proceedings of the 2nd ACM Conference on Electronic Commerce (EC)*, pp. 197–206. ACM (2000)
- [5] Chapelle, O., Schölkopf, B., Zien, A. (eds.): *Semi-Supervised Learning*. MIT Press (2006)
- [6] Cox, T.F., Cox, M.A.A.: *Multidimensional Scaling*. Chapman and Hall/CRC (2000)
- [7] Dinuzzo, F., Schölkopf, B.: The representer theorem for Hilbert spaces: A necessary and sufficient condition. In: *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*, pp. 189–196 (2012)
- [8] Endert, A., Han, C., Maiti, D., House, L., Leman, S., North, C.: Observation-level interaction with statistical models for visual analytics. In: *IEEE VAST*, pp. 121–130. IEEE (2011)
- [9] Forsythe, G.E., Golub, G.H.: On the Stationary Values of a Second-Degree Polynomial on the Unit Sphere. *Journal of the Society for Industrial and Applied Mathematics* 13(4), 1050–1068 (1965)
- [10] Gander, W.: Least Squares with a Quadratic Constraint. *Numerische Mathematik* 36, 291–308 (1981)
- [11] Gander, W., Golub, G., von Matt, U.: A constrained eigenvalue problem. *Linear Algebra and its Applications* 114–115, 815–839 (1989)
- [12] Ham, J., Lee, D.D., Mika, S., Schölkopf, B.: A Kernel View of the Dimensionality Reduction of Manifolds. In: *Proceedings of the 21st International Conference on Machine Learning* (2004)
- [13] Izenman, A.J.: *Linear Discriminant Analysis*. Springer (2008)
- [14] Jeong, D.H., Ziemkiewicz, C., Fisher, B.D., Ribarsky, W., Chang, R.: iPCA: An Interactive System for PCA-based Visual Analytics. *Comput. Graph. Forum* 28(3), 767–774 (2009)
- [15] Jolliffe, I.T.: *Principal Component Analysis*. Springer (1986)
- [16] Leman, S., House, L., Maiti, D., Endert, A., North, C.: Visual to Parametric Interaction (V2PI). *PLoS One* 8, e50474 (2013)
- [17] Li, R.C.: Solving secular equations stably and efficiently (1993)
- [18] Paurat, D., Gärtner, T.: Invis: A tool for interactive visual data analysis. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) *ECML PKDD 2013, Part III*. LNCS, vol. 8190, pp. 672–676. Springer, Heidelberg (2013)
- [19] Paurat, D., Oglic, D., Gärtner, T.: Supervised PCA for Interactive Data Analysis. In: *Proceedings of the Conference on Neural Information Processing Systems (NIPS) 2nd Workshop on Spectral Learning* (2013)
- [20] Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* 290(5500), 2323–2326 (2000)
- [21] Schölkopf, B., Herbrich, R., Smola, A.J.: A generalized representer theorem. In: Helmbold, D., Williamson, B. (eds.) *COLT/EuroCOLT 2001*. LNCS (LNAI), vol. 2111, pp. 416–426. Springer, Heidelberg (2001)
- [22] Shearer, C.: The CRISP-DM model: The new blueprint for data mining. *Journal of Data Warehousing* 5(4), 13–22 (2000)
- [23] Tenenbaum, J.B., Silva, V.D., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500), 2319–2323 (2000)
- [24] Tukey, J.W.: Mathematics and the picturing of data. In: *Proceedings of the International Congress of Mathematicians*, vol. 2, pp. 523–531 (1975)
- [25] Walder, C., Henao, R., Mørup, M., Hansen, L.K.: Semi-Supervised Kernel PCA. *Computing Research Repository (CoRR)* abs/1008.1398 (2010)
- [26] Weinberger, K.Q., Saul, L.K.: Unsupervised Learning of Image Manifolds by Semidefinite Programming. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 988–995 (2004)