4 rue Léonard de Vinci
BP 6759
F-45067 Orléans Cedex 2
FRANCE
http://www.univ-orleans.fr/lifo

Rapport de Recherche

Interactive and progressive constraint definition for dimensionality reduction and visualization

Lionel Martin, Matthieu Exbrayat,
Guillaume Cleuziou and Frédéric Moal
LIFO, Université d'Orléans

Rapport n^o **RR-2010-04**

Interactive and progressive constraint definition for dimensionality reduction and visualization

Lionel Martin, Matthieu Exbrayat, Guillaume Cleuziou and Frédéric Moal

May 18, 2010

Abstract

In this paper we focus on semi-supervised dimensionality reduction. Projecting and visualizing objects in a low dimension space is a current data analysis task. From another point of view, several works have been conducted recently, that constrain the projection space in order to optimize a given criteria (e.g. classification according to nearest neighbors). Nevertheless, none of those offers a satisfying interaction level. We thus propose an approach that offers to the user the ability to interactively add knowledge in the form of (di)similarity constraints among objects, when those appear either to close or to far in the current observation space. Such constraints can be added iteratively, the projection being modified after each new constraint. We propose several kinds of constraints and present a resolution method that derives from PCA.

Experiments have been performed with both synthetic and usual datasets. They show that a satisfying representation (w.r.t a given quality criterion) can be obtained with a small set of constraints.

1 Introduction

Machine learning techniques do generally aim at making out, according to a given objective, what is relevant in a given amount of data. Automated classification, both supervised and unsupervised, does especially fit to this definition. Objects to be classified are usually described by a -potentially large-set of features, which might be noisy, redundant, or simply non relevant according to the expected classification. Moreover, a large set of features might affect the readability of the classification criteria (as far as the underlying method offers readability).

Two kinds of approaches can be used to drop down the number of features. The first one, named *feature selection* [7], consists in choosing which features are the most relevant according to a given criteria. The second one, named *dimensionality reduction*, consists in combining the original features into a much smaller set of synthetic variables. These latter are computed in order to limit the distortion, in term of global distribution of objects, amongst the original and resulting spaces. We will hereafter use equivalently the terms of feature and variable. We should underline that dimensionality reduction is sometimes considered as being a subcase of feature selection (namely, a *filter method*) [3].

In this paper we will focus on dimensionality reduction methods, as they often lead to a much restricted set of features at the price of a reasonable lost of information. As a consequence, machine learning (distance-based comparison, estimation of gaussian mixtures,. . .) can be achieved at a lower cost, while visualization in a two- or three-dimensional space becomes available. Duality amongst classification and visualization is a remarkable aspect of dimensionality reduction: visualization can be considered as an intuitive tool, somehow open to non-expert users, while automated classification offers a quantitative and objective evaluation of the projection technique.

Many such approaches have been developed during the past decades, one of the most famous being *Principal Component Analysis* (PCA). Let us remind that, this unsupervised method does linearly combine the original features, in order to produce orthogonal dimensions that preserve the original variance at most. PCA is solved by identifying the most significant eigenvalues (and their associated eigenvectors) of the objects' covariance matrix in the original space. In the case of supervised classification, the label of training data will modify the criteria to be optimized. For instance, the *Linear Discriminant Analysis* (LDA, [6]) will project data in order to maximize the Fisher's criteria, i.e. the ratio of inter- and intra-class variance.

In some cases, data might spread along a given topological structure. Methods based on the concept of manifold have thus been developed. A first category of methods proceed in two steps: pairs of nearby objects are first identified; a projection is then computed, that optimizes a global criteria (e.g. variance) while limiting the distortion, in term of distance, for the pairs of nearby objects identified during the first step. Projection can be computed either by determining eigenvalues [9], or by solving a constrained system [8, 11]. A second group of methods uses a single step approach, such as the *Curvilinear Component Analysis* [5], which aims at minimizing the sum of differences between original and projected distance. A weight is associated to each pair of object. The closer the two points in the original space, the greater the weight affected to their pair. As a consequence, the projection will be distortion-tolerant for objects that are distant in the original space.

Concerning projection techniques, recent works have been focusing on semi-supervised methods, that imply distance criteria, either local or global, among objects that should belong to the same class or not. *Relevant Component Analysis* and *Large Margin Nearest Neighbors* can be considered as two representative proposals for this domain.

Relevant Component Analysis [2] consists in a semi-supervised technique. Only a small subset of objects is labeled, according to which the projection matrix has to be computed. To be more precise, the user only indicates pairs of objects that belong to the same class (i.e. *must-link* constraints). Using the transitive closure principle, the algorithm builds so-called *chunklets*, i.e. groups of objects of the same class (we must remind that these chunklets do only contain a small subset of the input set of objects X). An intra-chunklet covariance matrix $\hat{C}$ is built, that can either serve as a basis object projection ($X_{new} = \hat{C}^{-1/2}X$) or to compute a Mahalanobis distance $d(x_1, x_2) = (x_1 - x_2)^t \hat{C}^{-1}(x_1 - x_2)$. As a variant, an intermediate dimensionality-reduction step, based on $\hat{C}$, can e added.

Large Margin Nearest Neighbors (LMNN) is a supervised method that focuses on neighborhood [10, 12]. It does first identify the neighbors of each point, in the original space, according to a given radius. A set of constraints is then built, that expresses that each object should be closer to its same-class neighbors than from its different-class ones. Such a constraint is based on a minimum margin between the two kinds of neighbors. The authors implement this approach as a semi definite program, which produces a matrix M that serves as a basis for a Mahalanobis distance among objects. From M, a low-dimension projection matrix L can be computed such that $M = L^t L$.

Both RCA and LMNN show limitations. The number of constraints introduced in LMNN can be large. Solving them might be costly, especially if classes are mixed in (some parts of) the original space. RCA limits to must-link constraints, while other kinds of constraints could be used, that express that objects should be moved closer or away. Moreover, these techniques, as most of existing techniques, use a static set of constraints. No additional knowledge can be added during the dimensionality reduction step. We thus estimate that achieving both the simplicity of RCA and the power of LMNN would be worth, especially if we can also propose an intuitive and iterative way to inject knowledge .

In this paper we focus on semi-supervised learning, where objects are globally unlabeled. We propose three kinds of constraints, both intuitive and easy to use:

- relative position of two objects : inter object distance is bounded by a (upper or lower) limit;
- relative position of two objects (b, c) according to a third one (a): distance(a, c)/ distance(a,

b) is bounded.

- object neighborhood : a given object should be placed in the neighborhood of a given set of objects. This can be seen as a generalization of the former case.

It should be possible to add all of these constraints iteratively. According to a 2- or 3D projection, the user should be able to introduce constraints that indicate additional correction to the current projection. For instance, when two "similar" objects appear far away one from the other.

We must underline that this approach aims at using distance constraints in order to reduce dimensionality so that object should be comfortably viewed in a 2- or 3D space. It thus differs from RCA and LMNN, not only by its set of constraint types, but also by its obvious graphical interaction abilities. On the contrary, it differs from classical visual exploration tools, such as Grand Tour and Projection Pursuit [4], as it does not limit to moving the user's point of view. Projection is directly guided by the user.

This paper is organized as follows: in section 2, we formally describe the three kinds of constraints introduced. In section 3, we detail a solving mechanism based on the Uzawa algorithm. Validation tests and their results are presented and discussed in section 4. We sum up and propose various directions for further researches in section 5.

2 Constraints Formalization

Let a set of (observed) objects $x_1, ..., x_n$ described in $\mathbb{R}^d$, d being the dimensionality. In the remainder we will denote as $X = (x_1, ..., x_n)^T$ the matrix of observed objects, where row i contains the features describing object x_i. We aim at finding a k-dimensional representation of these objects, $k \ll d$, such that:

- information loss remains limited, which can be expressed as a maximal variance projection (as in PCA),
- user-specified constraints are satisfied *as much as possible*.

$k = 3$ will be of particular interest, in order to offer a 3D graphical interface.

This representation will be achieved through a projection in a k-dimension subspace ($k \ll d$). We must thus identify k vectors u_1, u_2 ... u_k associated with the projection matrix $L = (u_1, u_2, \dots u_k)$, such that the rows of $X.L$ correspond to the projections of the original objects in the k-dimension subspace. Let $h(x_i) = L^T.x_i$ the projection of object x_i. In this context, the (squared) euclidean distance among x_i and x_j *after projection*, $d^2(h(x_i)h(x_j))$, can be expressed as:

$$\begin{aligned} d^2(h(x_i),h(x_j)) &= < h(x_i)-h(x_j), h(x_i)-h(x_j) > && (1) \\ &= (h(x_i)-h(x_j))^T.(h(x_i)-h(x_j)) \\ &= (L^T.x_i - L^T.x_j)^T.(L^T.x_i - L^T.x_j) \\ &= (L^T.(x_i-x_j))^T.(L^T.(x_i-x_j)) \\ &= (x_i-x_j)^T.L.L^T.(x_i-x_j) && (2) \end{aligned}$$

In the remainder, we will denote $M = L.L^T$ as the distance matrix. We will aim at solving this dimensionality-reduction problem, while preserving variance at best, under the hereafter proposed constraints.

2.1 Relative Position of Two Objects ($C2$ Constraints)

Let us first consider the case, where a user wishes to modify the distance among two objects x_a and x_b. Let $\tilde{d}$ the expected distance after modification. Two subcases can be identified :

- ($C2_c$) x_a and x_b should be moved closer: $d^2(h(x_a),h(x_b)) \leq \tilde{d}$
- ($C2_a$) x_a and x_b should be moved away: $d^2(h(x_a),h(x_b)) \geq \tilde{d}$

Such constraints can be expressed by a quadruplet $(a,b,\tilde{d},\alpha)$, having $\alpha = 1$ for constraints of type $C2_c$, and $\alpha = -1$ for constraints of type $C2_a$. Thus, the constraint corresponding to $(a,b,\tilde{d},\alpha)$ expresses:

$$\alpha * [d^2(h(x_a),h(x_b)) - \tilde{d}] \leq 0 \quad (3)$$

The set of such constraints will be denoted as $C2$ in the remainder.

2.2 Relative Position of Two Objects Regarding a Third One ($C3$)

This second kind of constraint aims at modifying the relative position of a given object x_c with respect to two objects x_a and x_b. Two subcases might occur:

- x_c should be moved closer to x_a than x_b, so that $d^2(h(x_a),h(x_c)) \leq d^2(h(x_a),h(x_b))$. Let δ the strength of this ratio: $d^2(h(x_a),h(x_c)) \leq \delta * d^2(h(x_a),h(x_b))$
- x_b should be moved closer to x_a than x_c, so that $d^2(h(x_a),h(x_c)) \geq d^2(h(x_a),h(x_b))$. Introducing a strength δ we have: $d^2(h(x_a),h(x_c)) \geq \delta * d^2(h(x_a),h(x_b))$

Let $C3$ the set of such constraints. Such a constraint can be defined as a quintuplet $(a,b,c,\delta_{abc},\alpha)$, with $\alpha = 1$ for the first subcase, $\alpha = -1$ for the second one. Thus, if $(a,b,c,\delta,\alpha) \in C3$, the corresponding constraint can be expressed as:

$$\alpha * [d^2(h(x_a),h(x_c)) - \delta * d^2(h(x_a),h(x_b))] \leq 0 \quad (4)$$

2.3 Neighborhood Modification (Cn)

This third kind of constraints aims at modifying the neighborhood of a given object x_a. Let Nc_a the current neighborhood of x_a and Ne_a its expected neighborhood. In other terms, x_a should both get near the objects in Ne_a and away from those in Nc_a. This can be expressed through a set of constraints on x_a and objects in both neighborhoods.
Concerning Ne_a, the set of objects $k_a \in Ne_a$ is given by the user. Nc_a, the current neighborhood, consists of the k nearest neighbors of x_a in the source space. k can either be arbitrarily fixed (e.g. $k = 3$) or set to the value of k_a. For the sake of simplicity, we consider that $Nc_a \cap Ne_a = \emptyset$. Neighborhood modification constraints can be expressed as follows:

- $\forall x_i \in Nc_a$, let d_{ai} the observed distance, in the current space, between x_a and x_i. The modified distance is expected to be greater than the current one. This can be expressed as follows:
  $\forall x_i \in Nc_a \quad d^2(h(x_a),h(x_i)) \geq \beta * d_{ai}$, β being a parameter to be set (e.g. $\beta = 5$).

  Let $d(x_a,x_i)$, the distance between x_a and x_i in the *source* space (before any projection). By definition, this distance is an upper bound of $d(h(x_a),h(x_i))$. Our constraint can thus be modified to express that $d^2(h(x_a),h(x_i))$ should be close to $d^2(x_a,x_i)$: $\forall x_i \in Nc_a \quad d^2(h(x_a),h(x_i)) \geq \gamma * d^2(x_a,x_i)$,$\gamma \in [0,1]$ being a parameter to be set (e.g. 0.75). We will use this formulation hereafter.

- Concerning objects in Ne_a, a first approach would consist in minimizing their distance to x_a: $d^2(h(x_a),h(x_i))$. A second one would consist in giving a fixed upper bound on $d^2(h(x_a),h(x_i))$, using a constraint. We propose an upper bound based on the average distance $\overline{d_{Ne_a}}$ between objects in Ne_a :

  $\forall x_i \in Ne_a \quad d^2(h(x_a),h(x_i)) \leq \varepsilon * \overline{d_{Va}}$

  ε being a parameter to be set (e.g. 1.5). This kind of constraints is very similar to the one defined for objects in Nc_a. We will thus use it hereafter.

Let Cn the set of constraints of this third kind. A constraint in Cn is a quintuplet (a,i,d,θ,α) such that:

$$\alpha * [(d(h(x_a),h(x_i)) - \theta * d)] \leq 0 \tag{5}$$

with:

- $\theta = \gamma$, $\alpha = -1$ and $d = d(x_a,x_i)$ for current neighborhood constraints, and
- $\theta = \varepsilon$, $\alpha = 1$ and $d = \overline{d_{Ne_a}}$ for expected neighborhood constraints.

3 Constraints Solving

Our global optimization problem can be expressed as:

$$\begin{cases} Max_L \sum_{i,j}(x_i - x_j)^t.L.L^t.(x_i - x_j) \\ \forall i,j \quad < u_i, u_j >= \delta(i,j) \\ \forall (a,b,\tilde{d},\alpha) \in C2 \quad \alpha * [d^2(h(x_a),h(x_b)) - \tilde{d}] \leq 0 \\ \forall (a,b,c,\delta,\alpha) \in C3 \quad \alpha * [d^2(h(x_a),h(x_c)) - \delta * d^2(h(x_a),h(x_b))] \leq 0 \\ \forall (a,i,d,\theta,\alpha) \in Cn \quad \alpha * [d^2(h(x_a),h(x_i)) - \theta * d] \leq 0 \end{cases}$$

Each constraint uses $d^2(h(x_a),h(x_i))$, which depends on the problem solution (i.e. the matrix L). Once again, we can express this distance using the matrix formulation $d^2(h(x_a),h(x_i)) = (x_a - x_i)^t.L.L^t.(x_a - x_i)$ and note that:

$$\begin{aligned} d^2(h(x_a),h(x_i)) &= (x_a - x_i)^t.L.L^t.(x_a - x_i) \qquad (6) \\ &= \sum_{j=1..k} u_j^t.(x_a - x_i).(x_a - x_i)^t.u_j \qquad (7) \end{aligned}$$

where $(x_a - x_i).(x_a - x_i)^t$ is a $n \times n$ *matrix*. Let $X_{a,i}$ this matrix. Thus:
$d^2(h(x_a),h(x_i)) = \sum_{j=1..k} u_j^t.X_{a,i}.u_j$
Unlike the constraint-free case, an iterative search of $u_1, u_2 \ldots u_k$ does not apply any more, due to the *global* nature of user-defined constraints: these latter should not be achieved on each projection dimension independently, but rather globally in the projection space. Computing the k u_j vector must be processed simultaneously.
The criteria to be maximized can be expressed as:

$$\sum_{i,j}(x_i - x_j)^t.L.L^t.(x_i - x_j) = \sum_{j=1..k} u_j^t.X^t.X.u_j \tag{8}$$

Let us first consider the Lagrangian $\mathscr{L}$ of this problem, ignoring orthogonality constraints amongst vectors u_j:

$$\begin{aligned}
\mathscr{L}(L,\lambda,\mu,\psi,\rho) = & \sum_{j=1..k} u_j^t.X^t.X.u_j + \sum_{j=1..k} \lambda_j(u_j^t.u_j - 1) \\
+ & \sum_{(a_i,b_i,\tilde{d}_i,\alpha_i)\in C2} \mu_i * ((\sum_{j=1..k} u_j^t.X_{a_i,b_i}.u_j) - \tilde{d}_i) * \alpha_i) \\
+ & \sum_{(a_i,b_i,c_i,\delta_i,\alpha_i)\in C3} \psi_i * ((\sum_{j=1..k} u_j^t.X_{a_i,c_i}.u_j) \\
& \qquad - (\delta_i * (\sum_{j=1..k} u_j^t.X_{a_i,b_i}.u_j)) * \alpha_i) \\
+ & \sum_{(a_i,i_i,d_i,\theta_i,\alpha_i)\in Cn} \rho_i * (((\sum_{j=1..k} u_j^t.X_{a_i,i_i}.u_j) - \theta_* d_i) * \alpha_i)
\end{aligned}$$

Let us derive $\mathscr{L}$ according to u_j (to simplify the writing, we consider twice the derivative):

$$\begin{aligned}
2\frac{\partial \mathscr{L}(L,\lambda,\mu,\psi,\rho)}{\partial u_j} = & -X^t.X.u_j + \lambda_j.u_j + \sum_{(a_i,b_i,\tilde{d}_i,\alpha_i)\in C2} \mu_i \alpha_i X_{a_i,b_i}.u_j \\
+ & \sum_{(a_i,b_i,c_i,\delta_i,\alpha_i)\in C3} \psi_i \alpha_i * (X_{a_i,c_i}.u_j - \delta_i * (X_{a_i,b_i}.u_j)) \\
+ & \sum_{(a_i,i_i,d_i,\theta_i,\alpha_i)\in Cn} \rho_i \alpha_i * (X_{a_i,i_i}.u_j)
\end{aligned}$$

Let $\mathscr{X}_C$ such that:

$$\begin{aligned}
\mathscr{X}_C = & X^t.X - \sum_{(a_i,b_i,\tilde{d}_i,\alpha_i)\in C2} \mu_i \delta_i X_{a_i,b_i} \\
- & \sum_{(a_i,b_i,c_i,\delta_i,\alpha_i)\in C3} \psi_i \delta_i * (X_{a_i,c_i} - \delta_i * (X_{a_i,b_i})) \\
- & \sum_{(a_i,i_i,d_i,\theta_i,\alpha_i)\in Cn} \rho_i \alpha_i * (X_{a_i,i_i})
\end{aligned}$$

the partial derivative of $\mathscr{L}$ is then:

$$2\frac{\partial \mathscr{L}(L,\lambda,\mu,\psi,\rho)}{\partial u_j} = -\mathscr{X}_C.u_j + \lambda_j.u_j \tag{9}$$

We can notice that $\mathscr{L}$ can be expressed as:

$$\mathscr{L}(L,\lambda,\mu,\psi,\rho) = -\sum_{j=1..k} u_j^t \mathscr{X}_C.u_k + \sum_{j=1..k} \lambda_j(u_j^t.u_j - 1) \tag{10}$$

The partial derivative cancellation gives $\mathscr{X}_C.u_j = \lambda_j.u_j$. In other words, solutions u_j^* are eigenvectors of matrix $\mathscr{X}_C$, associated to eigenvalues λ_j. From this we can deduce two noticeable facts. First although no orthogonality constraint was expressed, the solution vectors u_j are orthogonal. Second, the dual function $q(\lambda,\mu,\psi,\rho)$ of our problem is:

$$q(\lambda,\mu,\psi,\rho) = Min_L \mathscr{L}(L,\lambda,\mu,\psi,\rho)$$

Indeed, using the optimality conditions defined above:

$$\begin{aligned}
q(\lambda,\mu,\psi,\rho) &= Min_L - \sum_{j=1..k} u_j^t \mathscr{X}_C.u_j + \sum_{j=1..k} \lambda_j(u_j^t.u_j - 1) \\
&= Min_L - \sum_{j=1..k} u_j^T.(\lambda_j u_j) + \sum_{j=1..k} \lambda_j(u_j^t.u_j - 1) \\
&= Min_L - \sum_{j=1..k} \lambda_j ||u_j||^2 + \sum_{j=1..k} \lambda_j(||u_j||^2 - 1) \\
&= Min_L - \sum_{j=1..k} \lambda_j
\end{aligned}$$

$||u_j||^2 = 1$ being a constraint for the optimal solution.
Last, as the dual problem consists in maximizing the dual function, the optimal solution corresponds to maximizing the sum of the k first eigenvalues of matrix $\mathscr{X}_C$, i.e. the λ_j corresponding to them. Nevertheless, to compute these eigenvalues we should give the value of the dual variables μ, ψ and ρ, whereas they are currently unknown. Due to this, we propose to compute these latter by the mean of the Uzawa iterative algorithm. We present this algorithm in the next section.

3.1 The Uzawa Algorithm

Uzawa algorithm was first introduced by [1]. Its main idea consists in determining a saddle point of the Lagrangian by the mean of successive approximations. Basically, this algorithm considers the following optimization problem:

$$\begin{cases} Min_x J(x) \\ h(x) = 0 \\ g(x) \leq 0 \end{cases}$$

where h and g do actually refer to families of functions h_i and g_j. Its Lagrangian is thus:

$\mathscr{L}(x,\lambda,\mu) = J(x) + \sum_i \lambda_i h_i(x) + \sum_j \mu_j g_j(x)$

Uzawa algorithm consists in setting the initial values of Lagrange coefficients (λ^0, μ^0), then computing the Lagrangian optimum, then modifying the coefficients according to this solution, and so forth. It iterates until convergence (which is guaranteed).

```
1 - Set $\lambda^0, \mu^0$,
2 - Iterate for $n \geq 0$ ($\rho$ is a parameter):
    2.1 Compute solution $x_n$ for: $Min_x L(x, \lambda^n, \mu^n)$
    2.2 Update $(\lambda^n, \mu^n)$ so that:
        $\lambda_i^{n+1} = \lambda_i^n + \rho * h_i(x_n)$
        $\mu_j^{n+1} = max(0, \mu_j^n + \rho * g_j(x_n))$
    2.3 Check for convergence: $||x_{n+1} - x_n|| < \varepsilon$
```

3.2 Implementation of the Uzawa Algorithm

In the context of our optimization problem, the Uzawa algorithm can be slightly simplified. λ_i do not need to be approximated: the x_n solutions being there the eigen vectors $u_1 \dots u_k$, the eigen values are a direct consequence of them. We will thus only set values for coefficients μ^n, i.e., in our case, μ and ψ.

Let $\mu^0 = \psi^0 = 0$. The first iteration consists in a classical PCA, as $\mathscr{X}_C = X^t.X$,

Afterward, unsatisfied constraints will be used to modify the matrix $\mathscr{X}_C$: according to Uzawa algorithm, if ($g_j(x_n) \leq 0$) is not satisfied, then $g_j(x_n) > O$, which implies the following update: $\mu_j^{n+1} = max(0, \mu_j^n + \rho * g_j(x_n))$, in order to verify $\mu_j^{n+1} > 0$. $\mathscr{X}_C$ will thus be computed according to

$g_j(x_n)$. This update of $\mathscr{X}_C$ is rather intuitive: let us consider a $C2$ constraint: $d^2(h(x_a),h(x_b)) \geq \tilde{d}$. If this constraint is not satisfied, when computing $\mathscr{X}_C$, we should add $cX_{a,b}$ (c being a constant) to $X^T.X$. Noticing that $X_{a,b}$ is already present in $X^T.X$, this update can be viewed as increasing the "weight" of $d^2(h(x_a),h(x_b))$ in the (unconstrained) criteria to be optimized. This weight will be increasing as long as the constraint remains unsatisfied...

To sum up, $\mathscr{X}_C$ is updated at each iteration by adding to it a set of matrices $cX_{a,b}$, and then diagonalized.

We should underline that this approach allows for "soft constraints", were convergence can be achieved having some constraints still unsatisfied. This is of interest in the case of hard-to-achieve, on even contradictory constraints.

4 Experiments

One of the main goals of the method we introduce consists in providing a graphical, interactive tool, for which users can iteratively add constraints and visualize their impact on the distribution of objects in the projection space. Nevertheless, the objective efficiency of a graphical tool is in essence difficult to evaluate. We thus propose a validation protocol in order to evaluate whether a satisfying projection can be achieved given a reasonably small number of constraints.

The criterion we propose consists in comparing the (3D) representation obtained with a reference one. Let us consider a set of objects. Intuitively, a user will consider a projection as satisfying if objects, that are very similar from her point of view, are projected nearby in the 3D space. In other words, if there exists a relevant classification of the objects, then objects of the same class will tend to be closer than objects of different classes. Fisher's criterion is a very well known criterion that precisely expresses this ratio: $\frac{\text{intra}-\text{class variance}}{\text{inter}-\text{classes variance}}$. Moreover, there exists a well known linear projection method, that optimizes this criterion, i.e. the Linear Discriminant Analysis (LDA). As a consequence, we could consider a LDA to produce a reference 3D organization of objects, regarding a relevant criterion.

We must underline that our projection method is not supervised in the sense that the projection mechanism is not aware of an existing classification of objects. It is semi-supervised in the sense that the user introduces constraints, that might be an expression of a class property, which is only materialized in the user's mind. We thus propose the following protocol: The raw input dataset consists of objects, described by a set of attributes together with a class attribute.

From this raw dataset we build an unlabeled one, i.e. we keep the description attributes while removing the class's one. This unlabeled dataset will serve as the input of our projection method.

Second, we simulate a user. This user is supposed to have some knowledge on the similarity of objects. In a real case, this knowledge might be materialized by additional attributes, such as pictures. In our simulated case, this knowledge comes from the class attribute. Our user is also supposed to be able to observe some strong distortion between her classification knowledge and the observed projection. We can suppose he would first try to overcome these strong distortions. The last point we have to make clear consists in identifying these distortions. Fortunately, the LDA can be considered as a very good tool, as it produces an optimal projection according to Fisher's criterion. For each pair of objects, we can compare its distance both in our projection and in the LDA one and then obtain a sorted list of their distortions. The user will then identify the most noticeable distortion, and introduce a (set of) constraint(s) in order to correct it.

Once again, the class knowledge is only used to simulate the user's way of highlighting distortions, not as a direct input for our algorithm.

Let d_{lda} the distance in the 3D space corresponding to the projection according to the three most significant dimensions of the LDA (the user being supposed to work in a 3D space). In the next section we introduce the various kinds of constraints we propose to study.

4.1 Constraints Generation

Let us remind that our initial projection corresponds to a PCA. Let $d(a,b)$ the distance between objects a and b after projection. We will implicitly consider we work on a three dimension projection. We propose to study five kinds of constraints:

$C2_{inf}$: in order to generate a $d(a,b) \leq \tilde{d}$ constraint, we must choose a pair of objects a and b along with a threshold distance value $\tilde{d}$. As we expect to move closer together a and b (more precisely, their projections in the subspace), we should choose objects that are currently projected far away from each other, while close according to d_{lda}. Thus, the bigger distortion corresponds to the pair such that $\frac{d(a,b)}{d_{lda}(a,b)}$ is maximal. The threshold distance is then $\tilde{d} = d_{lda}(a,b)$;

$C2_{sup}$: Similarly, we can introduce a constraint $d(a,b) \geq d_{lda}(a,b)$, considering the pair of objects that minimizes $\frac{d(a,b)}{d_{lda}(a,b)}$;

$C3_{lda}$: C_3 constraints express as $d(a,c) \leq \delta d(a,b)$. We thus have to choose three objects along with a threshold. We propose to generate such constraints triples a,b,c such that, a and c belong to the same class, and b belongs to another one. The goal thus consists in moving c (projected) close to a (projected), depending on its distance to b. We thus choose the triple a,b,c that maximizes $\frac{d(a,c)}{d(a,b)}$. The threshold is set to $\delta = \frac{d_{lda}(a,c)}{d_{lda}(a,b)}$;

$C3_1$: Rather than setting δ according to the LDA, we set it to 1. The corresponding constraint is then $d(a,c) \leq d(a,b)$, i.e. c should be closer to a than b is to a. The triple a, b, c is chosen in the same way as for $C3_{lda}$;

$C3_{1/2}$: δ is set to 1/2. This aims at observing a clear separation of classes.

In our experiments, each test does exclusively correspond to one of these categories of constraints. Constraints are introduced one by one according to the choice criterion defined above. We first start from a PCA projection. We choose the first constraint according to that projection and then compute a new projection. The second constraint is chosen according to this new projection, and so on. We thus simulate a user that introduces a constraint, observes its effects, then add a second one, and so forth.

4.2 Evaluation of Inertia on a Synthetic Dataset

We first consider a synthetic dataset, consisting of 75 objects. Each object corresponds to a word described by 48 attributes: (14 syntactic ones, 4 categorical ones, 20 semantic ones and 10 noise attributes). What makes this dataset interesting is the fact that various classifications can be achieved (syntactic, semantic, categorical). Here we will consider the categorical classification. Words are grouped into four disjoint classes, these classes being mostly induced by the 20 sementic attributes.

Our quality criterion is: $\mathrm{Q} = \frac{\text{inter-class variance}}{\text{total variance}}$. The higher its value, the more moved away the classes. This criterion corresponds to the 3D projection along the three most significant orthogonal dimensions (which corresponds to a good visual organization).

Figure 1 presents the evolution of Q depending on the type and number of constraints. The upper horizontal line corresponds to the value of Q obtained with LDA, i.e. the optimal value (94.2%).

We can notice that most of constraint categories improve this ratio compared to PCA, except $C2_{sup}$. Best results are achieved by C_{2inf} constraints. Figure 2 presents the visual distribution of the synthetic dataset resulting from 0, 5, 30 and 100 $C2_{inf}$ constraints respectively. The underlying constraints have been generated according to the formerly described protocol. To make this distribution more understandable, each object has been colored and shaped according to the class it belongs to.

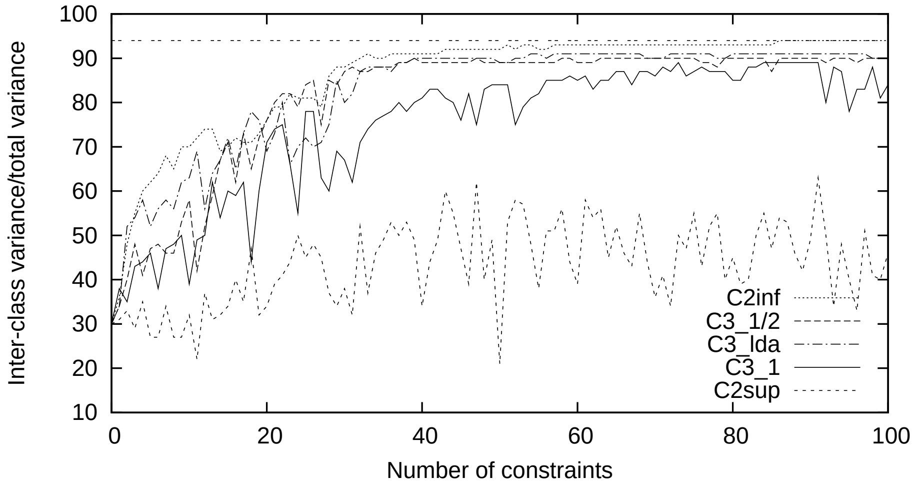


Figure 1: $\frac{inter-class\ variance}{total\ variance}$ depending on the number of constraints

We can observe that classes do move fast away as constraints are added. Concerning this dataset, thirty constraints are enough to reach a near-optimal ratio. Defining one-by-one such a number of constraints remains reasonable for a human user. Moreover, constraint solving is achieved in less than one second, even with the case of one hundred constraints.

Concerning $C2_{sup}$, this kind of constraints tends to move objects away, and should improve the ratio. Nevertheless, no noticeable gain is achieved, and the ratio remains highly unstable, even with a large number of constraints. Several elements can help to understand this counterperformance. First, the original *PCA* maximizes the global variance. As a consequence, most of the work consisting in moving objects away from each other has already been done. On the contrary, $C2_{inf}$ constraints (and others) introduce constraints that tend to move objects closer, which is a criterion that was out of the scope of *PCA*. Second, viewing the 3D projection highlights the impact of $C2_{sup}$ constraints. Figure 3 presents the visual distribution of the synthetic dataset resulting from 50 and 300 $C2_{sup}$ constraints respectively. Once again, the underlying constraints have been generated according to the former protocol, and for the sake of readibility, each object has been colored and shaped according to the class it belongs to. One can notice that using 50 $C2_{sup}$ constraints tends to globally stretch the underlying classes along axes that come through or near the origin. Considering that most of constraints concern objects that do not belong to the same class, we can easily see that objects will organize according to a set of axis as “orthogonal” one from each other as possible. Classes will be somehow characterized according to these axes, and barycenters will be globally moved away. But the lack of contracting constraints such as $C2_{inf}$ will tend to let objects spread along the axis and mix nearby the origin. With a larger set of constraints (i.e. 300) the graphical organization becomes readable. Neverthless, observing the $\frac{inter-class\ variance}{total\ variance}$ ratio, we can notice that this latter punctually reaches 80 %, but still oscillates mostly between 30 and 70 % when the number of constraints grows up to 300 and above.

We suggest that $C2_{sup}$ constraints might be more of interest in the case of datatets containing many attributes that, while not consisting of noise, are still irrelevant regarding the classification task, and thus the spatial distribution expected. In this case, moving objects away could help in projecting objects in a manner that deeply differs from the one of *PCA*.

As best results are achieved by C_{2inf} constraints, remaining tests will be based on this kind of constraint.

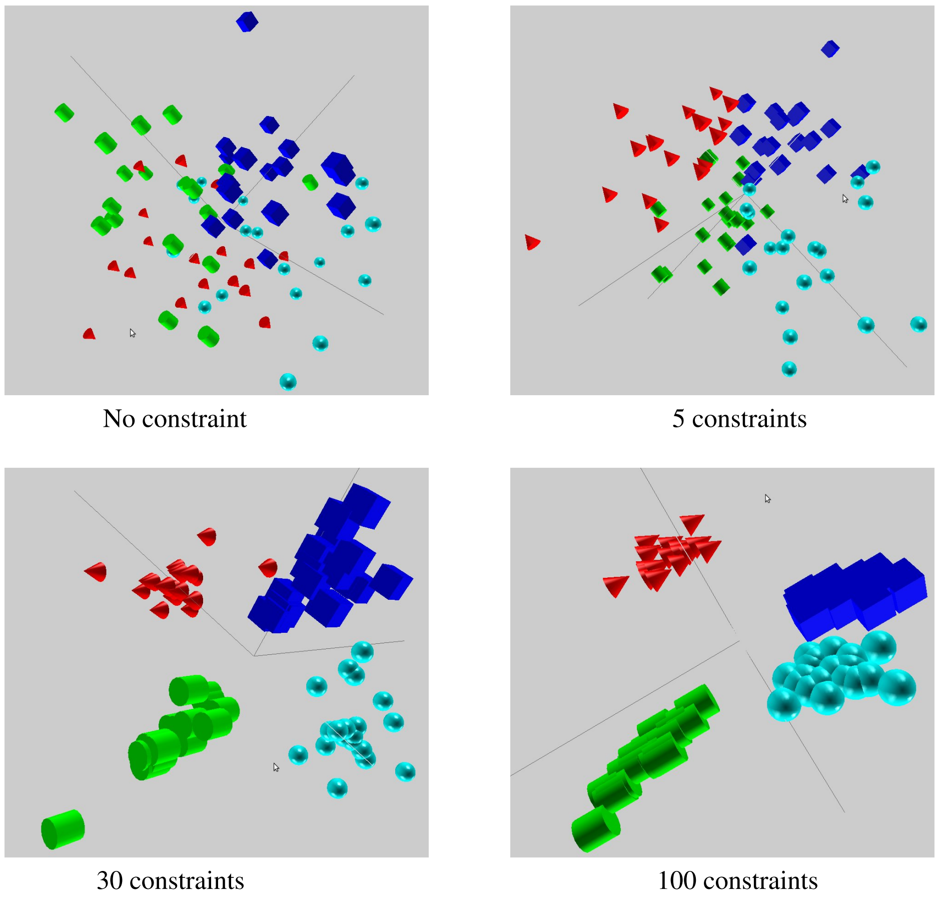
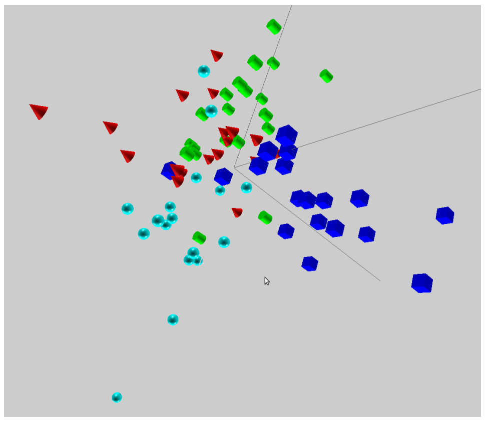
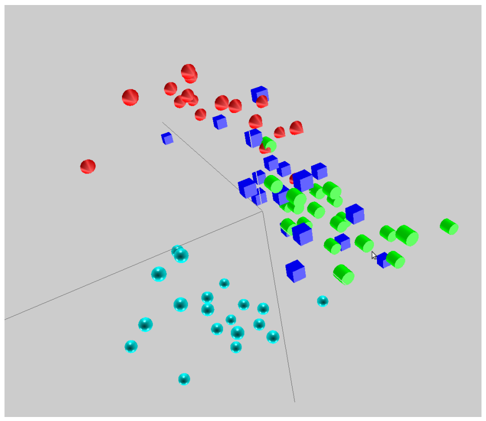


Figure 2: 3D projections with a growing number of $C2_{inf}$ constraints



50 constraints (highlight: classes' stretch)



300 constraints (classes remain confused)

Figure 3: 3D projection with $C2_{sup}$ constraints

4.3 Evaluation of Inertia on Standard Datasets

We conducted the same kind of tests on six classical machine learning datasets, namely *breast cancer*, *glass*, *iris*, *wine*, *yeast* and *zoo*, available at the *UCI repository*. Figure 4 presents the value of Q depending on the number of $C2_{inf}$ constraints. We can observe that these results are similar to the ones observed with the synthetic dataset.

In order to present these results on a single figure, all values are relative to the quality value Q_{LDA} achieved using *LDA*. Curves thus correspond to the evolution of the ratio Q/Q_{ALD}, a value near 100% thus corresponding to a value of the fisher criterion very similar to the one achieved with *LDA*.

The curve profiles are quite similar to the one of the synthetic dataset. For most of datasets, a optimal value of the quality criterion is reached using a very reduced set of constraints, except *breast cancer*, where a set of about thirty constraints is required. An anomaly seems to occur with *glass* where the quality criterion is above Q_{LDA}, whereas this latter is theoretically optimal. This can be explained by computation limits. The total variance is here very low with *LDA* (0.18 vs 1.89 with our approach). Thus the projection obtained trough *LDA* corresponds to a very low global dispersion, and might be inaccurate due to the effect of floating point calculus on the matrix inversion process.

4.4 Flexibility of our Approach

In the former sections, the simulated user does systematically choose to put a constraint on the higher distorsion observed w.r.t. the *LDA* projection. Such a strict strategy is likely to be inapplicable in the case of a real world user. Figure 5 presents the evolution of $\frac{inter-class\ variance}{total\ variance}$ with a relaxed strategy : at each step, the user does not choose the higher distorsion, but the hundredth one in decreasing order. We can observe that results remain quite similar to the ones on Figure 4. According to these results, a real world user will reach a good projection as long as she chooses interesting distorsions rather than the strictly most interesting ones w.r.t. her quality criterion (which might also not be clearly formalized).

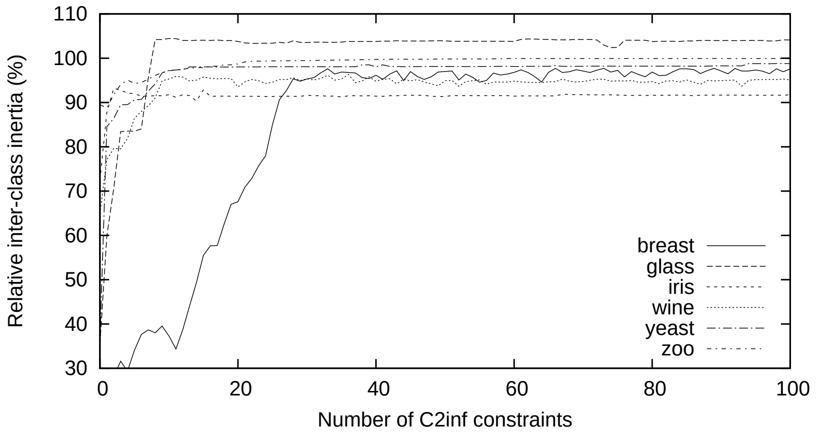


Figure 4: Evolution of $\frac{inter-class\ variance}{total\ variance}$ (UCI datasets)

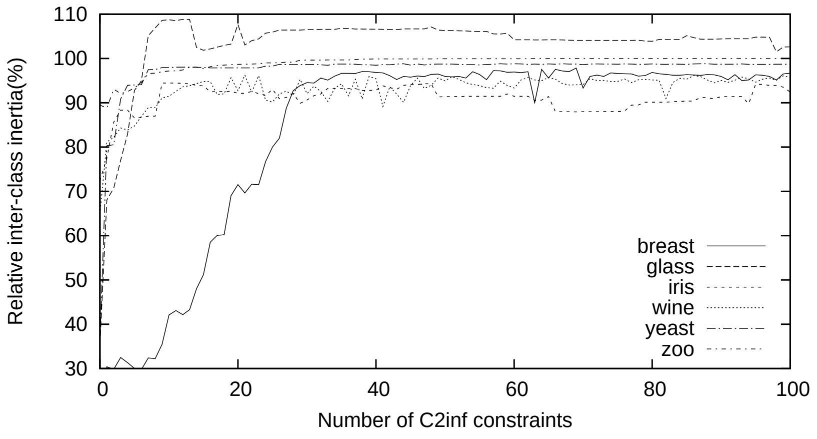


Figure 5: Evolution of $\frac{inter-class\ variance}{total\ variance}$ (smoth constraint choice algorithm)

4.5 Additional Experiments

Here we propose two additional experiments based on the UCI datasets, concerning how distorsions evolve while adding constraints. Figure 6 presents the evolution of the sum of observed distorsions w.r.t. the *ALD* projection. As datasets are very different, this sum differs greatly among them, and we thus use a log scale for the Y-axis. We can globally observe that, this sum decreases fast and then remains stable after a number of constraints that seems proportional to the initial sum of distorsions. One could thus consider that this initial sum is a good indicator of the number of constraints to define. However, in the case of a real, semi-supervised use, there is no formal definition of the optimal projection, and thus no way to automatically estimate (the initial sum of) distorsions. Nevertheless, from this experiment we can estimate that a consistent choice of a small set of constraints will tend to reduce the overall distorsions. We must underline that results using the relaxed strategy introduced in Sect. 4.4 are similar to those presented here (based on the strict choice of the higher distorsion at each step).

Figure 7 highlights the fast decreasing of distorsions when constraints are added. In order to draw

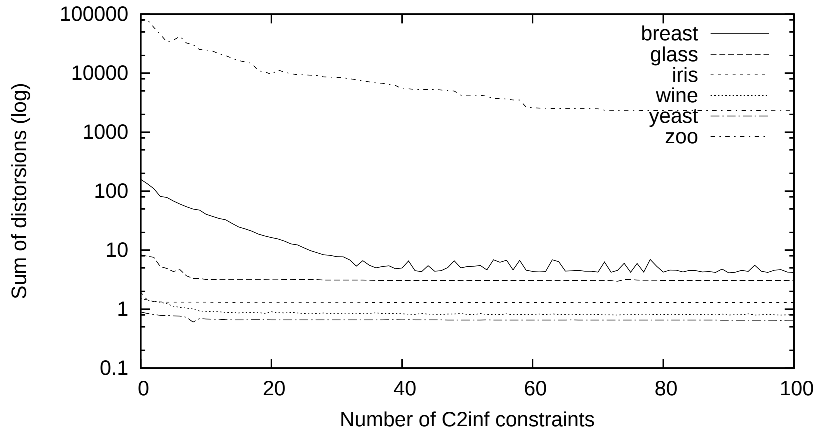


Figure 6: Evolution of the sum of distorsions

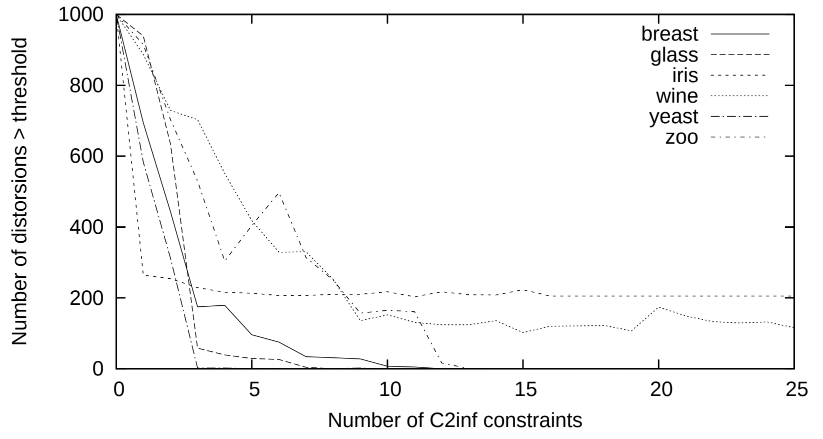


Figure 7: Evolution of the number of strong distorsions

this curve, we first computed the initial set of distorsions (without constraints), and kept the value of the thousandth one in decreasing order. We thus observed, after each addition of a constraint, the number of distorsions the value of which remained over this threshold. We can observe that this number does globally decrease fast. We can thus estimate that each new constraint has a great global influence on distorsions. Once again, we obtained similar results using the relaxed strategy.

5 Summary

In this paper we proposed a dimensionality reduction method, that allows an iterative and interactive definition of object positioning in the projection space. We observed that, for various datasets, adding a rather small set of constraints can lead to a satisfying projection, according to a user-given criteria. As a consequence, we estimate that such a method could be considered as a good way to diffuse dimensionality reduction techniques to a much wider population of non-specialist end users. Moreover, we introduce the concept of a progressive definition of constraints, and show how to solve it. As this concept helps to limit the size of the constraint set, we consider it as a promising approach.

Computing time is an important aspect of an interactive tool. We must notice that, while using a standard laptop computer, processing remained quite fast, usually much order one second.

This work is part of a French government *ANR* research grant. One goal of this project consists in proposing a visual organization of various middle age writing styles. In this context, each object consists in a picture (containing a sample writing), processed as a set of descriptors, these latter being automatically extracted from the picture. Most of these pictures are not associated to a class label, but paleographers are able to quantify the similarity between two writing styles. A validation with these experts is ongoing. Some additional types of constraints could be consequently introduced.

References

[1] Arrow, K., Hurwicz, L., Uzawa, H.: Studies in Nonlinear Programming. Stanford University Press, Stanford, CA (1958)

[2] Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning a mahalanobis metric from equivalence constraints. Journal of Machine Learning Research 6, 937–965 (2005)

[3] Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence 97, 245–271 (1997)

[4] Da Costa, D., Venturini, G.: A visual and interactive data exploration method for large data sets and custering. In: Springer (ed.) ADMA2007. pp. 553–561. No. 4632 in LNAI, Harbin, China (aug 2007)

[5] Demartines, P., Hrault, J.: Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. IEEE Transaction on Neural Networks 8(1), 148–154 (jan 1997)

[6] Fisher, R.: The use of multiple measurements in taxonomic problems. Annals of Eugenics 7, 179–188 (1936)

[7] Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)

[8] Roweis, S.T., Saul, L.K.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science 290(5500), 2323–2326 (2000)

[9] Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (December 2000)

[10] Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: NIPS (2005)

[11] Weinberger, K.Q., Saul, L.K.: Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision 70(1), 77–90 (2006)

[12] Weinberger, K.Q., Saul, L.K.: Fast solvers and efficient implementations for distance metric learning. In: Cohen, W.W., McCallum, A., Roweis, S.T. (eds.) ICML. ACM International Conference Proceeding Series, vol. 307, pp. 1160–1167. ACM (2008)