Eigenproblems in Pattern Recognition

Tijl De Bie[1], Nello Cristianini[2], and Roman Rosipal[3]

[1] K.U.Leuven ESAT-SCD/SISTA
Kasteelpark Arenberg 10
3001 Leuven, Belgium
tijl.debie@esat.kuleuven.ac.be

[2] U.C. Davis Department of Statistics
360 Kerr Hall One Shields Ave.
Davis, CA 95616
nello@support-vector.net

[3] NASA Ames Research Center
Computational Sciences Division
Moffett Field, CA 94035
rrosipal@mail.arc.nasa.gov

1 Introduction

The task of studying the properties of configurations of points embedded in a metric space has long been a central task in pattern recognition, but has acquired even greater importance after the recent introduction of kernel-based learning methods. These methods work by virtually embedding general types of data in a vector space, and then analyzing the properties of the resulting data cloud. While a number of techniques for this task have been developed in fields as diverse as multivariate statistics, neural networks, and signal processing, many of them show an underlying unity. In this chapter we describe a large class of pattern analysis methods based on the use of generalized eigenproblems, which reduce to solving the equation $\mathbf{Aw} = \lambda\mathbf{Bw}$ with respect to $\mathbf{w}$ and λ.

The problems in this class range from finding a set of directions in the data-embedding space containing the maximum amount of variance in the data (principal components analysis), to finding a hyperplane that separates two classes of data minimizing a certain cost function (Fisher discriminant), or finding correlations between two different representations of the same data (canonical correlation analysis). Also some important clustering algorithms can be reduced to solving eigenproblems. The importance of this class of algorithms derives from the facts that generalized eigenproblems provide an efficient way to optimize an important family of cost functions, of the type $f(\mathbf{w}) = \frac{\mathbf{w}'\mathbf{Aw}}{\mathbf{w}'\mathbf{Bw}}$ (known as a *Rayleigh quotient*); they can be studied with very simple linear algebra; and they can be solved or approximated efficiently using a number of well-known techniques from computational algebra.

Their statistical behavior has also been studied to some extent (e.g. [24] and [25]), allowing us to efficiently design regularization strategies in order to reduce the risk of overfitting. However, methods limited to detecting linear relations among vectors could hardly be considered to constitute state-of-the-art technology, given the nature of the challenges presented by modern data analysis. Therefore it is crucial that *all such problems* can be cast and solved in a kernel-induced feature space; that is, they only require information about inner products between data points. The entire toolbox of generalized eigenproblems for pattern analysis can then be applied to detection of generalized relations on a wide range of data types, such as sequences, text, images, and so on.

In this chapter we will first review the general theory of eigenvalue problems, then we will give a brief review of kernel methods in general. Finally, we will discuss a number of algorithms based in multivariate statistics: principal components analysis, partial least squares, canonical correlation analysis, Fisher discriminant, and spectral clustering, where appropriate both in their primal and in their dual form, leading to a version involving kernels.

1.1 Notation

All matrices are boldface uppercase. Vectors are boldface lowercase. Scalar variables are lowercase. Sets and spaces are denoted with calligraphic letters.

With $\left(\mathbf{a}\ \mathbf{b} \cdots \mathbf{z}\right)$, the matrix built by stacking the vectors $\mathbf{a}, \mathbf{b}, \ldots, \mathbf{z}$ next to each other is meant.

The symbols used are:

- The vector containing all ones is denoted by $\mathbf{1}$. The identity matrix is denoted by $\mathbf{I}$. The matrix or vector containing all zeros is denoted by $\mathbf{0}$. Their dimensionality is clear from the context.
- $\mathbf{x}$ or $\mathbf{x}_i$, column vectors represent a vector in the $\mathcal{X}$-space. When we have n samples, the matrix $\mathbf{X}$ is built up as $\mathbf{X} = \left(\mathbf{x}_1\ \mathbf{x}_2 \cdots \mathbf{x}_n\right)'$.
- Similarly, $\mathbf{y}$ or $\mathbf{y}_i$ are sample vectors from the $\mathcal{Y}$-space. The matrix $\mathbf{Y}$ containing samples $\mathbf{y}_1$ through $\mathbf{y}_n$ is built up as $\mathbf{Y} = \left(\mathbf{y}_1\ \mathbf{y}_2 \cdots \mathbf{y}_n\right)'$.
- When $\mathcal{Y}$ is one-dimensional, a sample from this space is denoted by y or y_i, and the vector containing all samples is $\mathbf{y} = \left(y_1\ y_2 \cdots y_n\right)'$.
- Unless stated differently, *all data are assumed to be centered (have zero mean) throughout this chapter*. This means that $\mathbf{1}' \cdot \mathbf{X} = \mathbf{0}'$, $\mathbf{1}' \cdot \mathbf{Y} = \mathbf{0}'$, or when $\mathcal{Y}$ is one-dimensional, $\mathbf{1}' \cdot \mathbf{y} = 0$.
- $\mathbf{K_X}$ and $\mathbf{K_Y}$ are the so-called *kernel* or *Gram matrices* corresponding to $\mathbf{X}$ and $\mathbf{Y}$. They are the inner product matrices $\mathbf{K_X} = \mathbf{XX}'$ and $\mathbf{K_Y} = \mathbf{YY}'$. When it is clear from the context which data the kernel is built from, we just use $\mathbf{K}$. When we want to stress the kernel is centered we use $\mathbf{K}_\mathrm{c}$.
- For centered data matrices $\mathbf{X}$ and $\mathbf{Y}$, the matrices $\mathbf{S_{XX}} = \mathbf{X}'\mathbf{X}$, $\mathbf{S_{XY}} = \mathbf{X}'\mathbf{Y}$, $\mathbf{S_{YY}} = \mathbf{Y}'\mathbf{Y}$, and $\mathbf{S_{YX}} = \mathbf{S_{XY}}'$ are the *scatter matrices.*

- $\boldsymbol{\alpha}, \boldsymbol{\alpha}_{\mathbf{X}}, \boldsymbol{\alpha}_{\mathbf{Y}}, \alpha_i, \alpha_{\mathbf{X},i}$, and $\alpha_{\mathbf{Y},i}$ will be referred to as *dual vectors* and their respective ith coordinates. When an index i is used as a subscript after a boldface $\boldsymbol{\alpha}$, this refers to a dual vector indexed by i, and not to the ith coordinate.
- $\mathbf{w}, \mathbf{w}_{\mathbf{X}}, \mathbf{w}_{\mathbf{Y}}$ will be referred to as *weight vectors*. Their respective ith coordinates are denoted by $w_i, w_{\mathbf{X},i}, w_{\mathbf{Y},i}$. When an index i is used as a subscript after a boldface $\mathbf{w}$, this refers to a weight vector indexed by i, and not to the ith coordinate.
- The feature map from the input space to the feature space is denoted with $\phi(\mathbf{x}_i)$.
- $d, n, m, \ldots$ are scalar integers; d is used for indicating dimensionality.

2 Linear Algebra

In this section we will review some basic properties of linear algebra that will prove useful in this chapter. We use the standard linear algebra notation in the beginning and translate the important results to the kernel methods conventions afterwards. Extensive references for matrix analysis can be found in [12] and [13].

2.1 Symmetric (Generalized) Eigenvalue Problems

Notation. In this introductory section, we will use a notation that is to be distinguished from the notation in the remainder of the chapter:

- $\mathbf{A} \in \mathcal{R}^{n \times m}$, a general matrix.
- $\mathbf{M}, \mathbf{N} \in \mathcal{R}^{n \times n}$, symmetric matrices. $\mathbf{N}$ is invertible.
- $\boldsymbol{\Lambda}, \mathbf{S} \in \mathcal{R}^{n \times n}$, diagonal matrices.
- $\mathbf{U}, \mathbf{V} \in \mathcal{R}^{n \times n} : \mathbf{U}\mathbf{U}' = \mathbf{I} = \mathbf{U}'\mathbf{U}, \mathbf{V}\mathbf{V}' = \mathbf{I} = \mathbf{V}'\mathbf{V}$, orthogonal matrices.
- $\mathbf{W} \in \mathcal{R}^{n \times n}$, a matrix orthogonal in the metric defined by $\mathbf{N}$: $\mathbf{w}'\mathbf{N}\mathbf{w} = \mathbf{I}$.
- λ or λ_i, an eigenvalue.
- σ or σ_i, a singular value.

2.1.1 Variational Characterization

The optimization problems we are concerned with in this chapter are all basically of the form (we assume $\mathbf{N}$ is invertible)

$$\max_{\mathbf{w}} \frac{\mathbf{w}'\mathbf{M}\mathbf{w}}{\mathbf{w}'\mathbf{N}\mathbf{w}}.$$

This is an optimization of a Rayleigh quotient. One can see the norm of $\mathbf{w}$ does not matter: scaling $\mathbf{w}$ does not change the value of the object function. Thus, one can impose an additional scalar constraint on $\mathbf{w}$ and optimize the object function without losing any solutions. This constraint is chosen to be

$\mathbf{w}'\mathbf{N}\mathbf{w} = 1$. Then the optimization problem becomes a constrained optimization problem of the form:

$$\max_{\mathbf{w}} \ \mathbf{w}'\mathbf{M}\mathbf{w} \quad \text{s.t.} \quad \mathbf{w}'\mathbf{N}\mathbf{w} = 1,$$

or by using the Lagrangian $\mathcal{L}(\mathbf{w})$:

$$\max_{\mathbf{w}} \ \mathcal{L}(\mathbf{w}) = \max_{\mathbf{w}} \mathbf{w}'\mathbf{M}\mathbf{w} - \lambda \mathbf{w}'\mathbf{N}\mathbf{w}.$$

Equating the first derivative to zero leads to

$$\mathbf{M}\mathbf{w} = \lambda \mathbf{N}\mathbf{w}. \tag{1}$$

The optimal value reached by the object function is equal to the maximal eigenvalue, the Lagrange multiplier λ. This is the symmetric generalized eigenvalue problem that will be studied here.

Note that the vector $\mathbf{w}$ with the scalar λ leading to the optimum of the Rayleigh quotient is not the only solution of the generalized eigenvalue problem given by Eq. (1). There exist other eigenvector–eigenvalue pairs that do not correspond to the optimum of the Rayleigh quotient. For any pair $(\mathbf{w}, \lambda)$ that is a solution of Eq. (1), $\mathbf{w}$ is called a (generalized) eigenvector and λ is called a (generalized) eigenvalue. In many cases several of these eigenvector–eigenvalue pairs are of interest.

2.1.2 Symmetric Eigenvalue Problems

For the ordinary symmetric eigenvalue problem (where $\mathbf{N} = \mathbf{I}$):

$$\mathbf{M}\mathbf{w} = \lambda \mathbf{w}.$$

Eigenvectors $\mathbf{w}_i$ corresponding to different eigenvalues λ_i are orthogonal to each other. Furthermore, the eigenvalues of symmetric matrices are real, and a real eigenvector corresponds to them.

Proof. For $\lambda_i \neq \lambda_j$,

$$\begin{aligned}
\mathbf{M}\mathbf{w}_i &= \lambda_i \mathbf{w}_i, \\
\Rightarrow \lambda_i(\mathbf{w}_j'\mathbf{w}_i) &= \mathbf{w}_j'\mathbf{M}\mathbf{w}_i = \mathbf{w}_i'\mathbf{M}'\mathbf{w}_j = \mathbf{w}_i'\mathbf{M}\mathbf{w}_j, \\
&= \lambda_j(\mathbf{w}_i'\mathbf{w}_j), \\
\Rightarrow \mathbf{w}_j'\mathbf{w}_i &= 0.
\end{aligned}$$

Thus, eigenvectors corresponding to different eigenvalues λ_i and λ_j are orthogonal. Furthermore, with $\cdot^*$ the adjoint operator:

$$\begin{aligned}
\mathbf{M}\mathbf{w}_i &= \lambda_i \mathbf{w}_i \;\text{ and }\; \mathbf{M} = \mathbf{M}' = \mathbf{M}^* \;\; (\mathbf{M} \text{ is real symmetric}) \;, \\
\Rightarrow \lambda_i^* \mathbf{w}_i' \mathbf{w}_i^* &= (\lambda_i \mathbf{w}_i^{*\prime} \mathbf{w}_i)^* = (\mathbf{w}_i^{*\prime} \mathbf{M}\mathbf{w}_i)^* = \mathbf{w}_i' \mathbf{M}^* \mathbf{w}_i^* = \mathbf{w}_i' \mathbf{M}' \mathbf{w}_i^*, \\
&= \lambda_i \mathbf{w}_i' \mathbf{w}_i^*, \\
\Rightarrow \lambda_i &= \lambda_i^*.
\end{aligned}$$

Therefore the eigenvalues of a real symmetric matrix are real. Then also the eigenvectors are real up to a complex scalar (and can thus be made real by scalar multiplication), since if they were not, we could take the real part and the imaginary part separately, and both would be eigenvectors corresponding to the same eigenvalue.

When eigenvalues are *degenerate*, that is, they are equal but correspond to a different eigenvector, then these eigenvectors can be chosen to be orthogonal to each other. This follows from the fact that they are in a subspace orthogonal to the space spanned by all eigenvectors corresponding to the other eigenvalues. In this subspace an orthogonal basis can be found. The number of eigenvalues and corresponding orthogonal eigenvectors of a real symmetric matrix thus is equal to the dimensionality n of $\mathbf{M}$.

If we normalize all eigenvectors $\mathbf{w}_i$ to unit length and choose them to be orthogonal to each other, they are said to form an *orthonormal* basis. For $\mathbf{W}$ being the matrix built by stacking these normalized eigenvectors $\mathbf{w}_i$ next to each other, we have

$$\mathbf{W}\mathbf{W}' = \mathbf{W}'\mathbf{W} = \mathbf{I},$$

that is, the matrix $\mathbf{W}$ is *orthogonal*.

Since then $\mathbf{M}\mathbf{w}_i = \mathbf{w}_i \lambda_i$ for all i, we can state that

$$\mathbf{M}\mathbf{W} = \mathbf{W}\boldsymbol{\Lambda},$$

where $\boldsymbol{\Lambda}$ contains the corresponding eigenvalues λ_i on its diagonal. Then, taking into account that $\mathbf{W}^{-1} = \mathbf{W}'$, we can express the matrix $\mathbf{M}$ as:

$$\mathbf{M} = \mathbf{W}\boldsymbol{\Lambda}\mathbf{W}' = \sum_i \lambda_i \mathbf{w}_i \mathbf{w}_i'.$$

This is called the *eigenvalue decomposition* of the matrix $\mathbf{M}$, also known as the *spectral decomposition* of $\mathbf{M}$.

2.1.3 Symmetric Generalized Eigenvalue Problems

In general, we will deal with generalized eigenvalue problems of the form

$$\mathbf{M}\mathbf{w} = \lambda \mathbf{N}\mathbf{w}.$$

This could be solved as an ordinary but nonsymmetric eigenvalue problem (by multiplying with $\mathbf{N}^{-1}$ on the left-hand side). We can also convert it to a symmetric eigenvalue problem by defining $\mathbf{v} = \mathbf{N}^{1/2}\mathbf{w}$:

$$\mathbf{M}\mathbf{N}^{-1/2}\mathbf{N}^{1/2}\mathbf{w} = \lambda\mathbf{N}^{1/2}\mathbf{N}^{1/2}\mathbf{w},$$

and thus by left multiplication with $\mathbf{N}^{-1/2}$:

$$(\mathbf{N}^{-1/2}\mathbf{M}\mathbf{N}^{-1/2})\mathbf{v} = \lambda\mathbf{v}.$$

For this type of problem, we know that the different eigenvectors $\mathbf{v}$ can be chosen to be orthogonal and of unit length, thus:

$$\mathbf{V}'\mathbf{V} = \mathbf{I} = \mathbf{W}'\mathbf{N}\mathbf{W},$$

which means that the generalized eigenvectors $\mathbf{w}_i$ of a symmetric eigenvalue problem are orthogonal in the metric defined by $\mathbf{N}$.

2.2 Singular Value Decompositions, Duality

The singular value decomposition of a general real matrix $\mathbf{A}$ is defined as

$$\mathbf{A} = \begin{pmatrix} \mathbf{U} & \mathbf{U}_0 \end{pmatrix} \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{V} & \mathbf{V}_0 \end{pmatrix}' = \mathbf{U}\mathbf{S}\mathbf{V}',$$

where $\mathbf{S}$ contains the *singular values* s_i in decreasing order (by convention) on the diagonal, and dimensions of all blocks are compatible. The matrices $\begin{pmatrix} \mathbf{U} & \mathbf{U}_0 \end{pmatrix}$ and $\begin{pmatrix} \mathbf{V} & \mathbf{V}_0 \end{pmatrix}$ are orthogonal matrices, respectively containing the *left* and the *right singular vectors* as their columns. This decomposition can be calculated for any real matrix.

One can see that multiplying $\mathbf{A}$ on the left with a column of $\mathbf{U}_0$ gives zero: $\mathbf{U}_0'\mathbf{A} = \mathbf{0}'$. Therefore $\mathbf{U}_0$ is said to span the left *null space* of $\mathbf{A}$. Similarly, $\mathbf{V}_0$ is a basis for the right null space of $\mathbf{A}$. On the other hand, $\mathbf{U}$ and $\mathbf{V}$ respectively span the column and the row space of $\mathbf{A}$.

Note that $\mathbf{A}\mathbf{A}'$ and $\mathbf{A}'\mathbf{A}$ are symmetric, and their eigenvalue decompositions are:

$$\begin{aligned} \mathbf{A}\mathbf{A}' &= \mathbf{U}\mathbf{S}^2\mathbf{U}', \\ \mathbf{A}'\mathbf{A} &= \mathbf{V}\mathbf{S}^2\mathbf{V}'. \end{aligned}$$

Another important property of singular value decompositions is that the nonzero singular values and corresponding singular vectors are the nonzero eigenvalues and corresponding eigenvectors of the matrix $\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}' & \mathbf{0} \end{pmatrix}$:

$$\begin{pmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A}' & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_i \\ \mathbf{v}_i \end{pmatrix} = s_i \begin{pmatrix} \mathbf{u}_i \\ \mathbf{v}_i \end{pmatrix}, \tag{2}$$

the solution of which leads to the singular value decomposition of $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}'$.

In a pattern recognition problem, the rows of the matrix $\mathbf{A}$ may consist of different data vectors. Above, we used the standard linear algebra notation. In pattern recognition, the matrix $\mathbf{A}$ will then correspond to $\mathbf{X}$, the columns of $\mathbf{V}$ to $\mathbf{w}$ being the weight vectors, and the columns of $\mathbf{U}$ to α, being the dual vectors. Thus, in the notation we adopt in this chapter:

$$\begin{aligned} \mathbf{X}'\alpha_i &= s_i\mathbf{w}_i, \\ \mathbf{X}\mathbf{w}_i &= s_i\alpha_i. \end{aligned}$$

When the norm is not an issue, which is often the case, the factor s_i can be omitted, so up to a scaling factor:

$$\begin{aligned} \mathbf{X}'\alpha_i &= \mathbf{w}_i, \\ \mathbf{X}\mathbf{w}_i &= \alpha_i. \end{aligned} \tag{3}$$

The matrix $\mathbf{X}'\mathbf{X} = \mathbf{S}_{\mathbf{XX}}$ will be called a scatter matrix. Since the samples making up the rows of $\mathbf{X}$ are assumed to have zero mean, it is proportional to the finite sample covariance matrix $\mathbf{C}_{\mathbf{XX}} = \frac{1}{n}\mathbf{S}_{\mathbf{XX}}$. On the other hand, $\mathbf{X}\mathbf{X}' = \mathbf{K}_{\mathbf{X}}$ is a Gram or kernel matrix . (Note that element (i, j) corresponds to the inner product of samples $\mathbf{x}_i$ and $\mathbf{x}_j$.) Thus, the weight vectors are the eigenvectors of the scatter matrix, and the dual vectors are the eigenvectors of the kernel matrix. Given the dual vectors, the weight vectors can be found by multiplication with the data matrix $\mathbf{X}'$, and vice versa. *This type of relation between primal and dual variables forms the basis of the duality and enables the use of kernels.*

3 Kernel Methods

Kernel methods (KMs) [7, 21, 23, 27, 29] are a relatively new family of algorithms that presents a series of useful features for pattern analysis in data sets. In recent years, their simplicity, versatility, and efficiency have made them a standard tool for practitioners, and a fundamental topic in many data analysis courses. We will outline some of their important features, referring the interested reader to more detailed articles and books for a deeper discussion (see, for example, [23] and references therein).

KMs combine the simplicity and computational efficiency of linear algorithms, such as the perceptron algorithm or ridge regression, with the flexibility of nonlinear systems, such as, for example, neural networks, and the rigor of statistical approaches, such as regularization methods in multivariate statistics. As a result of the special way they represent functions, these algorithms typically reduce the learning step to a simple optimization problem that can always be solved in polynomial time, avoiding the problem of local minima typical of neural networks, decision trees, and other nonlinear approaches.

Their foundation in the principles of statistical learning theory makes them remarkably resistant to overfitting especially in regimes where other methods are affected by the 'curse of dimensionality'. Another important feature for applications is that they can naturally accept input data that are not in the form of vectors, such as, for example, strings, trees, and images. Their characteristically modular design makes them amenable to theoretical analysis, but also makes them well suited to a software engineering approach in which a general-purpose learning module is combined with a data-specific 'kernel function' that provides the interface with the data and incorporates domain knowledge.

Many learning modules can be used, depending on whether the task is one of classification, regression, clustering, novelty detection, ranking, and so on. At the same time, many kernel functions have been designed, for example, for protein sequences, for text and hypertext documents, for images, time series, etc. As a result, this method can be used for dealing with rather exotic tasks, such as ranking strings, or clustering graphs, in addition to such classical tasks as classifying vectors. In the remainder of this section, we will briefly describe theory behind kernel methods, followed by a brief example of how this can be used in practice: kernelizing least squares regression and ridge regression.

3.1 Theory

Kernel-based learning algorithms work by embedding the data into a Hilbert space and searching for linear relations in such space. The embedding is performed implicitly, that is, by specifying the inner product between each pair of points, rather than by giving their coordinates explicitly. This approach has several advantages, the most important being the observation that often the inner product in the embedding space can be computed much more easily than the coordinates of the points themselves.

Given an input set $\mathcal{X}$ and an embedding vector space $\mathcal{F}$ (often called the feature space), we consider a map $\phi : \mathcal{X} \to \mathcal{F}$ (often called the feature map). The function that, given two points $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{x}_j \in \mathcal{X}$, returns the inner product between their images in the space $\mathcal{F}$ is known as *kernel function.*

Definition 1. *A* kernel *is a function k, such that for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$, $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle$, where ϕ is a mapping from $\mathcal{X}$ to a Hilbert space $\mathcal{F}$, and $\langle \cdot, \cdot \rangle$ denotes the inner product.*

We also consider the matrix $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, called the *kernel matrix* or the *Gram matrix*. Thanks to the fact it is built from inner products it is always a symmetric, positive semidefinite matrix, and since it specifies the inner products between all pairs of points, it completely determines the relative positions between those points in the embedding space. For example, given such information, it is trivial to recover all the pairwise distances between them.[1]

The solutions sought by kernel-based algorithms are linear functions in the feature space:

$$f(\mathbf{x}) = \mathbf{w}'\phi(\mathbf{x}),$$

for some weight vector $\mathbf{w}$. The kernel can be exploited whenever the weight vector can be expressed as a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i)$, implying that we can express f as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}).$$

This will be the case for any of the algorithms considered in this chapter.

3.2 Example: Least Squares and Ridge Regression

We consider the well-known problem of least squares regression to start with and derive a *kernelized* version for it. Consider the vector $\mathbf{y} \in \mathcal{R}^n$ and the data points $\mathbf{X} \in \mathcal{R}^{n \times d}$. We want to find the weight vector $\mathbf{w} \in \mathcal{R}^d$ that minimizes $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$. Taking the gradient of this cost function with respect to $\mathbf{w}$ and equating to zero leads to:

$$\begin{aligned} \nabla_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 &= \nabla_{\mathbf{w}}(\mathbf{y}'\mathbf{y} + \mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w} - 2\mathbf{w}'\mathbf{X}'\mathbf{y}), \\ &= 2\mathbf{X}'\mathbf{X}\mathbf{w} - 2\mathbf{X}'\mathbf{y}, \\ &= 0, \\ \Rightarrow \mathbf{w} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}. \end{aligned}$$

This is the well-known least squares solution.

However, least squares is highly sensitive to overfitting. Especially when $\mathbf{X}$ lives in a high-dimensional (feature) space, care needs to be taken (ultimately, when the dimensionality $d > n$, regression can always be carried out exactly, which means that any noise sequence could be fit by the model). In order to avoid overfitting, a standard approach is to reduce the *capacity* of the learner, or the *effective number of degrees of freedom*, by imposing a prior on the solution, thus introducing a bias. In the case of regression, for example, one usually prefers a weight vector with small norm. This is taken into account by introducing an additional term $\gamma\|\mathbf{w}\|^2$ in the cost function, with γ the *regularization parameter*. Minimizing leads to the ridge regression estimate:

[1] Notice that we do not really need $\mathcal{X}$ to be a vector space; in fact, $\mathcal{X}$ can be a generic finite set. This is because we are guaranteed that the data are *implicitly* mapped to some Hilbert space by simply checking that the kernel matrix $\mathbf{K}$ satisfies the conditions above.

$$\begin{aligned}\nabla_{\mathbf{w}}\left[\|\mathbf{y}-\mathbf{X}\mathbf{w}\|^2+\gamma\|\mathbf{w}\|^2\right] &= \nabla_{\mathbf{w}}\left[(\mathbf{y}'\mathbf{y}+\mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w}-2\mathbf{w}'\mathbf{X}'\mathbf{y})+\gamma(\mathbf{w}'\mathbf{w})\right],\\ &= 2(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})\mathbf{w}-2\mathbf{X}'\mathbf{y},\\ &= 0,\\ \Rightarrow \mathbf{w} &= (\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})^{-1}\mathbf{X}'\mathbf{y}.\end{aligned}$$

To evaluate the regression function in a new test point, it can simply be projected on the weight vector:

$$y_{\text{test}} = \mathbf{x}'_{\text{test}}\mathbf{w}.$$

So far we have discussed the primal version of the ridge regression method. The dual version can be derived by noting that the minimum norm weight vector will always be in the span of the data $\mathbf{X}$. This can be seen by replacing $(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I})^{-1}$ with $(\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}'+\gamma\mathbf{I})^{-1} = (\mathbf{V}(\boldsymbol{\Lambda}+\gamma\mathbf{I})\mathbf{V}')^{-1} = \mathbf{V}(\boldsymbol{\Lambda}+\gamma\mathbf{I})^{-1}\mathbf{V}'$, where the columns of $\mathbf{V}$ are the right singular vectors of $\mathbf{X}$ and are thus a basis for the row space of $\mathbf{X}$. Thus the weight vector $\mathbf{w} = \mathbf{V}\left[(\boldsymbol{\Lambda}+\gamma\mathbf{I})^{-1}\mathbf{V}'\mathbf{X}'\mathbf{y}\right]$ lies in the column space of $\mathbf{V}$, or equivalently in the row space of $\mathbf{X}$, and can thus be expressed as $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha}$ (cf. Eq. (3)). Here $\boldsymbol{\alpha} \in \mathcal{R}^n$ is called the dual vector. Plugging this into the equations leads to:

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}}\left[\|\mathbf{y}-\mathbf{X}\mathbf{X}'\boldsymbol{\alpha}\|^2+\gamma\|\mathbf{X}'\boldsymbol{\alpha}\|^2\right] &= 2(\mathbf{X}\mathbf{X}'\mathbf{X}\mathbf{X}')\boldsymbol{\alpha}-2\mathbf{X}\mathbf{X}'\mathbf{y}+2\gamma\mathbf{X}\mathbf{X}'\boldsymbol{\alpha},\\ &= 2(\mathbf{K}^2+\gamma\mathbf{K})\boldsymbol{\alpha}-2\mathbf{K}\mathbf{y},\\ &= \mathbf{0},\\ \Rightarrow \mathbf{K}(\mathbf{K}+\gamma\mathbf{I})\boldsymbol{\alpha} &= \mathbf{K}\mathbf{y}. \qquad (4)\end{aligned}$$

In the second step, $\mathbf{X}\mathbf{X}'$, which is the matrix containing the inner products between any two points as its elements, is replaced by the kernel matrix $\mathbf{K}$. Since the inner products in $\mathbf{K}$ can be inner products in a feature space, they can in fact be a nonlinear function of the data points, namely the kernel function. In this way, nonlinearities can be dealt with in a very natural way. This is the essence of the 'kernel trick'. A general solution for Eq. (4) is given by:

$$\boldsymbol{\alpha} = (\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y}+\boldsymbol{\alpha}_0,$$

where $\boldsymbol{\alpha}_0$ is any vector in the null space of $\mathbf{K}$: $\mathbf{X}'\boldsymbol{\alpha}_0 = \mathbf{K}\boldsymbol{\alpha}_0 = \mathbf{0}$.

The projection of a test point $\mathbf{x}_{\text{test}}$ onto the weight vector $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha} = \mathbf{X}'\left[(\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y}+\boldsymbol{\alpha}_0\right] = \mathbf{X}'(\mathbf{K}+\gamma\mathbf{I})^{-1}\mathbf{y}$, can be written as $y_{\text{test}} = \mathbf{x}'_{\text{test}}\mathbf{X}'\boldsymbol{\alpha}$ (as one can see, the actual value of $\boldsymbol{\alpha}_0$ does not matter). Written in terms of kernel evaluations, this becomes:

$$y_{\text{test}} = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}).$$

This is indeed the standard form.

3.3 Kernels in This Chapter

In this chapter, we will aim at deriving primal and dual versions of spectral algorithms in pattern recognition. Whereas the primal formulation is usually the standard form in which algorithms are known, the dual form is formulated in terms of inner products only.[2] This is important, since then the kernel trick can be used in any algorithm where such a dual version can be derived, very much in the same way as shown in the example above: by replacing the matrix containing inner products with the kernel matrix. The inner products are considered to be carried out implicitly between nonlinear mappings of the points in a feature space.

As mentioned before, we will assume all data are centered. In primal space, this centering is a trivial operation, as it is done by simply subtracting the mean of each of the coordinates (n is the number of samples): $\mathbf{X}_c = \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)$. However, centering in feature space deserves some attention since we do not compute the feature vectors explicitly, but only the inner products between them. Thus we have to compute the centered kernel matrix based on the uncentered kernel matrix.

For an uncentered $\mathbf{K}$ corresponding to uncentered $\mathbf{X}$, the centered version $\mathbf{K}_{\mathrm{c}}$ can be computed as the product of the centered matrices $\mathbf{X}_{\mathrm{c}} = \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)$, where $\mathbf{1} \in \mathcal{R}^n$ is the column vector containing n ones:

$$\begin{aligned}\mathbf{K}_{\mathrm{c}} &= \left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)\left(\mathbf{X} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{X}\right)' \\ &= \mathbf{K} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K} - \mathbf{K}\frac{\mathbf{1}\mathbf{1}'}{n} + \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K}\frac{\mathbf{1}\mathbf{1}'}{n}. \end{aligned} \tag{5}$$

In this chapter, unless stated otherwise, we assume all kernel matrices are centered as such. Therefore, the subscript c will be omitted for brevity, wherever this does not cause confusion.

Similarly, a test sample $\mathbf{x}_{\mathrm{test}}$ should be centered accordingly. Let $\mathbf{k}_{\mathrm{test}} = [k(\mathbf{x}_{\mathrm{test}}, \mathbf{x}_i)]_{i=1:n}$ be the vector containing the kernel evaluations of $\mathbf{x}_{\mathrm{test}}$ with all n training samples $\mathbf{x}_i$. Then again, we can do the centering implicitly: the properly centered version (in correspondence with the centering of Eq. (5)) of this vector can be shown to be

$$\mathbf{k}_{\mathrm{test,c}} = \mathbf{k}_{\mathrm{test}} - \mathbf{K}\frac{\mathbf{1}}{n} - \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{k}_{\mathrm{test}} + \frac{\mathbf{1}\mathbf{1}'}{n}\mathbf{K}\frac{\mathbf{1}}{n}.$$

In this chapter we assume all test samples are already centered in this way as well. Again, the subscript c will be omitted wherever this does not cause confusion.

[2] In many if not all practical cases, the dual can be motivated using an optimization perspective. The reader is referred to [27] for an in-depth treatment.

4 Dimensionality Reduction: PCA, (R)CCA, PLS

The general philosophy that motivates dimensionality reduction techniques is the fact that real-life data contain redundancies and noise. Dimensionality reduction is often a good way to deal with this: by using a low-dimensional approximate representation, noise can be suppressed and redundancies removed. The data are replaced by a summary that still captures as much information as possible. All methods described in this section can be useful as a preprocessing step for other algorithms like clustering, classification, regression, and so on.

We will discuss various ways to perform dimensionality reduction. They all share the property that they rely on inner products and on eigenproblems. This has as a consequence that they can easily be made nonlinear using the kernel trick, and that they are efficiently solved. The difference between them lies in the cost function they optimize.

Therefore, each of the subsections will be structured as follows: first the different cost functions leading to the algorithm are described, subsequently the primal is derived and some properties are given, and finally the dual formulation is presented. For a previous treatment of these algorithms in their primal version, we refer to [6].

4.1 PCA

4.1.1 Cost Function

The motivation for performing *principal component analysis* (PCA) [16] is often the assumption that directions of high variance will contain more information than directions of low variance. The rationale behind this could be that the noise can be assumed to be uniformly spread. Thus, directions of high variance will have a higher signal-to-noise ratio. Mathematically:

$$\begin{aligned}\mathbf{w} &= \text{argmax}_{\|\mathbf{w}\|=1}\mathbf{w}'\mathbf{X}'(\mathbf{w}'\mathbf{X}')',\\ &= \text{argmax}_{\|\mathbf{w}\|=1}\mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w},\\ &= \text{argmax}_{\|\mathbf{w}\|=1}\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}. \qquad (6)\end{aligned}$$

Or, for $\mathbf{w}$ not normalized this can be written as:

$$\mathbf{w} = \text{argmax}_{\mathbf{w}}\frac{\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}}{\mathbf{w}'\mathbf{w}}.$$

The solution of Eq. (6) is also equivalent to minimizing the 2-norm of the residuals. This can be seen by projecting all samples $\mathbf{X}$ on the subspace orthogonal to $\mathbf{w}$ (by left multiplication with $(\mathbf{I}-\mathbf{w}\mathbf{w}')$), and computing the Frobenius norm:

$$
\begin{aligned}
\mathbf{w} &= \mathrm{argmin}_{\|\mathbf{w}\|=1}\|\mathbf{X}(\mathbf{I}-\mathbf{w}\mathbf{w}')\|_{\mathrm{F}}^2, \\
&= \mathrm{argmin}_{\|\mathbf{w}\|=1}\mathrm{trace}\left([\mathbf{X}(\mathbf{I}-\mathbf{w}\mathbf{w}')]'[\mathbf{X}(\mathbf{I}-\mathbf{w}\mathbf{w}')]\right), \\
&= \mathrm{argmin}_{\|\mathbf{w}\|=1}\mathrm{trace}\left(\mathbf{X}'\mathbf{X}+\mathbf{w}\mathbf{w}'\mathbf{X}'\mathbf{X}\mathbf{w}\mathbf{w}'-2\mathbf{X}'\mathbf{X}\mathbf{w}\mathbf{w}'\right), \\
&= \mathrm{argmin}_{\|\mathbf{w}\|=1}\mathrm{trace}(\mathbf{S}_{\mathbf{XX}})+\|\mathbf{w}\|^2\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}-2\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}, \\
&= \mathrm{argmin}_{\|\mathbf{w}\|=1}-\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}.
\end{aligned}
$$

4.1.2 Primal

Differentiating the Lagrangian $\mathcal{L}(\mathbf{w},\lambda)=\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}-\lambda\mathbf{w}'\mathbf{w}$ corresponding to Eq. (6) with respect to $\mathbf{w}$ and equating to zero leads to

$$
\begin{aligned}
&\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w},\lambda)=\nabla_{\mathbf{w}}(\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}-\lambda\mathbf{w}'\mathbf{w})=0, \\
\Leftrightarrow\ &\mathbf{S}_{\mathbf{XX}}\mathbf{w}=\lambda\mathbf{w}.
\end{aligned}
$$

This is a symmetric eigenvalue problem as presented in Sect. 2. Such an eigenvalue problem has d eigenvectors. All are called principal directions, corresponding to their variance λ.

Properties

- All principal directions are orthogonal to each other.
- The principal directions can all be obtained by optimizing the same cost function, where the above property is explicitly imposed.
- The projections of the data onto different principal directions are *uncorrelated*: $(\mathbf{X}\mathbf{w}_i)'\mathbf{X}\mathbf{w}_j=0$ for $i\neq j$. Note that one could as well say the projections are *orthogonal*. This is equivalent, but we will use the notion of correlation when we are talking about projections of data onto a weight vector. Because of this property of PCA, it is sometimes called *linear decorrelation*.
- The PCA solution is equivalent to, and can thus be obtained by computing, the singular value decomposition of $\mathbf{X}$.

4.1.3 Dual

To derive the dual, we use the key fact that $\mathbf{w}$ will always be a linear combination of the columns of $\mathbf{X}'$ (to see this, note that $\mathbf{w}=\frac{1}{\lambda}\mathbf{S}_{\mathbf{XX}}\mathbf{w}=\mathbf{X}'\frac{\mathbf{X}\mathbf{w}}{\lambda}$). We can thus replace $\mathbf{w}$ with $\mathbf{X}'\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ are the dual variables. The dual problem is then:

$$
\begin{aligned}
\mathbf{S}_{\mathbf{XX}}\mathbf{X}'\boldsymbol{\alpha}&=\lambda\mathbf{X}'\boldsymbol{\alpha}, \\
\Rightarrow\mathbf{X}\mathbf{S}_{\mathbf{XX}}\mathbf{X}'\boldsymbol{\alpha}&=\lambda\mathbf{X}\mathbf{X}'\boldsymbol{\alpha}, \\
\Rightarrow\mathbf{K}_{\mathbf{X}}^2\boldsymbol{\alpha}&=\lambda\mathbf{K}_{\mathbf{X}}\boldsymbol{\alpha}. \qquad (7)
\end{aligned}
$$

When $\mathbf{K_X}$ has full rank, we can multiply Eq. (7) by $\mathbf{K_X}^{-1}$ on the left-hand side, leading to:

$$\mathbf{K_X}\alpha = \lambda\alpha. \tag{8}$$

On the other hand, when $\mathbf{K_X}$ is rank deficient, a solution for Eq. (7) is not always a solution for Eq. (8) anymore (however, the converse is still true). Then for α_0 lying in the null space of $\mathbf{K_X}$, and α a solution of Eq. (8) (and thus also of Eq. (7)), also $\alpha + \alpha_0$ is a solution of Eq. (7) but generally not of Eq. (8). But, since $\mathbf{K_X}\alpha_0 = \mathbf{0}$ and thus $\mathbf{X}'\alpha_0 = \mathbf{0}$, the component α_0 will have no effect on $\mathbf{w} = \mathbf{X}'(\alpha + \alpha_0) = \mathbf{X}'\alpha$ anyway, and we can ignore the null space of $\mathbf{K_X}$ by simply solving Eq. (8) also in the case $\mathbf{K_X}$ is rank deficient.

Since $\mathbf{K_X}$ is a symmetric matrix, the dual eigenvectors will be orthogonal to each other. The projections of the training samples onto the weight vector $\mathbf{w}$ are $\mathbf{Xw} = \mathbf{XX}'\alpha = \lambda\alpha$. Thus, the vector α is proportional with (and thus up to a normalization equal to) the projections of the training samples onto this weight vector. The fact that different dual vectors are orthogonal is thus equivalent to the observation that the projections of the data onto different weight vectors is uncorrelated.

Projection of a test point onto the PCA direction found can be carried out as

$$y_{\text{test}} = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}}).$$

4.2 Canonical Correlation Analysis (CCA) and Regularized CCA

While PCA deals with only one data space $\mathcal{X}$ where it identifies directions of high variance, *canonical correlation analysis* (CCA, first introduced in [15]) proposes a way for dimensionality reduction by taking into account relations between samples coming from *two* spaces $\mathcal{X}$ and $\mathcal{Y}$. The assumption is that the data points coming from these two spaces contain some joint information that is reflected in correlations between them. Directions along which this correlation is high are thus assumed to be relevant directions when these relations are to be captured.

Again a primal and a dual form are available. The dual form makes it possible to capture nonlinear correlations as well, thanks to the kernel trick [1, 3, 11].

When data are scarce as compared to the dimensionality of the problem, it is important to regularize the problem in order to avoid overfitting. This is provided in the *regularized CCA* (RCCA) algorithm.

4.2.1 A Small Example

To make things more concrete, consider the following example described in [31]. Suppose we have two text corpora, one containing English texts, and another one containing the same texts but translated in French. The text corpora can be represented by the matrices $\mathbf{X}$ and $\mathbf{Y}$ containing vectors that are the bag of words representations of the texts as its rows. Now, since we know that the same basic semantic information must be present in both the English text and the French translation, we must be able to extract some information from every row of $\mathbf{X}$ that is similar to information extracted from the rows of $\mathbf{Y}$. If we do this in a linear way, this would mean that $\mathbf{X}\mathbf{w}_\mathbf{X}$ and $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ are similar in a way, for some $\mathbf{w}_\mathbf{X}$ and $\mathbf{w}_\mathbf{X}$ representing a certain semantic meaning. This could be: $\mathbf{X}\mathbf{w}_\mathbf{X}$ and $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ are correlated, thus motivating the cost function introduced below. In [31], it is pointed out that many of the $\mathbf{w}_\mathbf{X}$–$\mathbf{w}_\mathbf{X}$ pairs found can indeed be related to an intuitively satisfying semantic meaning. Other examples are available in literature, notably in bioinformatics [30, 35].

4.2.2 Cost Function

We thus want to maximize the correlation between a projection $\mathbf{X}\mathbf{w}_\mathbf{X}$ of $\mathbf{X}$ and a projection $\mathbf{Y}\mathbf{w}_\mathbf{Y}$ of $\mathbf{Y}$. Or, another geometrical interpretation is: find directions $\mathbf{X}\mathbf{w}_\mathbf{X}, \mathbf{Y}\mathbf{w}_\mathbf{Y}$ in the column space of $\mathbf{X}$ and $\mathbf{Y}$ with a minimal angle between each other (we will use the notation $\mathbf{S}_{\mathbf{XY}} = \mathbf{X}'\mathbf{Y}$, the *cross-scatter* matrix):

$$\begin{aligned}\{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}\} &= \text{argmax}_{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}} \cos\left(\angle(\mathbf{X}\mathbf{w}_\mathbf{X}, \mathbf{Y}\mathbf{w}_\mathbf{Y})\right), \\ &= \text{argmax}_{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}} \frac{(\mathbf{X}\mathbf{w}_\mathbf{X})'(\mathbf{Y}\mathbf{w}_\mathbf{Y})}{\sqrt{(\mathbf{X}\mathbf{w}_\mathbf{X})'(\mathbf{X}\mathbf{w}_\mathbf{X})}\sqrt{(\mathbf{Y}\mathbf{w}_\mathbf{Y})'(\mathbf{Y}\mathbf{w}_\mathbf{Y})}}, \\ &= \text{argmax}_{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}} \frac{\mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_\mathbf{Y}}{\sqrt{\mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}_\mathbf{X}}\sqrt{\mathbf{w}_\mathbf{Y}'\mathbf{S}_{\mathbf{YY}}\mathbf{w}_\mathbf{Y}}}.\end{aligned}$$

Since the norm of the weight vectors does not matter, we can maximize correlation along the weight vectors, or ‘fit’ subject to constraints fixing the value of these weight vectors:

$$\begin{aligned}\{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}\} &= \text{argmax}_{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}} \mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_\mathbf{Y} \\ &\qquad \text{s.t. } \|\mathbf{X}\mathbf{w}_\mathbf{X}\|^2 = \mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}_\mathbf{X} = 1, \|\mathbf{Y}\mathbf{w}_\mathbf{Y}\|^2 = \mathbf{w}_\mathbf{Y}'\mathbf{S}_{\mathbf{YY}}\mathbf{w}_\mathbf{Y} = 1.\end{aligned}$$

This is equivalent to the minimization of a ‘misfit’ subject to these constraints:

$$\begin{aligned}\{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}\} &= \text{argmin}_{\mathbf{w}_\mathbf{X}, \mathbf{w}_\mathbf{Y}} \|\mathbf{X}\mathbf{w}_\mathbf{X} - \mathbf{Y}\mathbf{w}_\mathbf{Y}\|^2 \\ &\qquad \text{s.t. } \|\mathbf{X}\mathbf{w}_\mathbf{X}\|^2 = 1, \|\mathbf{Y}\mathbf{w}_\mathbf{Y}\|^2 = 1.\end{aligned}$$

4.2.3 Primal

We solve the second formulation of the problem. Differentiating the Lagrangian $\mathcal{L}(\mathbf{w}_X, \mathbf{w}_\mathbf{Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = \mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_\mathbf{Y} - \lambda_\mathbf{X}\mathbf{w}_\mathbf{X}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}_\mathbf{X} - \lambda_\mathbf{Y}\mathbf{w}_\mathbf{Y}'\mathbf{S}_{\mathbf{YY}}\mathbf{w}_\mathbf{Y}$ with respect to $\mathbf{w}_\mathbf{X}$ and $\mathbf{w}_\mathbf{Y}$ and equating to 0, gives

$$\begin{cases} \frac{\partial}{\partial \mathbf{w_X}}\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = 0, \\ \frac{\partial}{\partial \mathbf{w_Y}}\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = 0, \end{cases}$$
$$\Rightarrow \begin{cases} \mathbf{S_{XY}w_Y} = \lambda_\mathbf{X}\mathbf{S_{XX}w_X}, \\ \mathbf{S_{YX}w_X} = \lambda_\mathbf{Y}\mathbf{S_{YY}w_Y}. \end{cases}$$

Now, since from this

$$\lambda_\mathbf{X}\mathbf{w'_X}\mathbf{S_{XX}w_X} = \mathbf{w'_X}\mathbf{S_{XY}w_Y} = \mathbf{w'_Y}\mathbf{S_{YX}w_X} = \lambda_\mathbf{Y}\mathbf{w'_Y}\mathbf{S_{YY}w_Y},$$

and since $\mathbf{w'_X}\mathbf{S_{XX}w_X} = \mathbf{w'_Y}\mathbf{S_{YY}w_Y} = 1$, we find that $\lambda_\mathbf{X} = \lambda_\mathbf{Y} = \lambda$, and thus

$$\begin{cases} \mathbf{S_{XY}w_Y} = \lambda\mathbf{S_{XX}w_X}, \\ \mathbf{S_{YX}w_X} = \lambda\mathbf{S_{YY}w_Y}. \end{cases} \tag{9}$$

Or, stated in another way as a generalized eigenvalue problem,

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}. \tag{10}$$

This generalized eigenvalue problem has $2d$ eigenvalues. But, for each positive eigenvalue λ and corresponding eigenvector $\begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}$, $-\lambda$ is an eigenvalue too with corresponding eigenvector $\begin{pmatrix} \mathbf{w_X} \\ -\mathbf{w_Y} \end{pmatrix}$. Thus, we get all the information by only looking at the d positive eigenvalues. The largest one with its eigenvector corresponds to the optimum of the cost function described earlier. The weight vectors making up the other eigenvectors will be referred to as other canonical directions, corresponding to a smaller canonical correlation quantized by their corresponding eigenvalue.

Properties

- CCA not only finds pairs of directions that capture maximal correlations between each other. Projections onto canonical directions corresponding to a different canonical correlation are *uncorrelated*:

$$\begin{aligned} \lambda_i \mathbf{w'_{Y,\mathit{j}}}(\mathbf{S_{YY}w_{Y,\mathit{i}}}) &= \mathbf{w'_{Y,\mathit{j}}}(\mathbf{S_{YX}w_{X,\mathit{i}}}), \\ &= \mathbf{w'_{X,\mathit{i}}}(\mathbf{S_{XY}w_{Y,\mathit{j}}}), \\ &= \lambda_j \mathbf{w'_{X,\mathit{i}}}(\mathbf{S_{XX}w_{X,\mathit{j}}}), \\ &= \lambda_j \mathbf{w'_{X,\mathit{j}}}(\mathbf{S_{XX}w_{X,\mathit{i}}}). \end{aligned}$$

  And similarly,

$$\lambda_i \mathbf{w'_{X,\mathit{j}}}(\mathbf{S_{XX}w_{X,\mathit{i}}}) = \lambda_j \mathbf{w'_{Y,\mathit{j}}}(\mathbf{S_{YY}w_{Y,\mathit{i}}}).$$

  So for $\lambda_i \neq \lambda_j$, the projection of $\mathbf{Y}$ onto $\mathbf{w_{Y,\mathit{j}}}$ is uncorrelated with the projection of $\mathbf{X}$ onto $\mathbf{w_{X,\mathit{i}}}$: $\mathbf{w'_{Y,\mathit{j}}}\mathbf{S_{YX}w_{X,\mathit{i}}} = 0$. Similarly, $\mathbf{w'_{X,\mathit{j}}}\mathbf{S_{XX}w_{X,\mathit{i}}} =$

0, and $\mathbf{w}'_{\mathbf{Y},j}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} = 0$. Another way to state this is to say that $\mathbf{w}_{\mathbf{X},i}$ is orthogonal to $\mathbf{w}_{\mathbf{X},j}$ in the metric defined by $\mathbf{S_{XX}}$; similarly, $\mathbf{w}_{\mathbf{Y},i}$ is orthogonal to $\mathbf{w}_{\mathbf{Y},j}$ in the metric defined by $\mathbf{S_{YY}}$.

- All canonical directions can be captured by a constrained optimization problem in which the above property is explicitly imposed:

$$\begin{aligned}\{\mathbf{w}_{\mathbf{X},i}, \mathbf{w}_{\mathbf{Y},i}\} &= \text{argmax}_{\mathbf{w}_{\mathbf{X},i},\mathbf{w}_{\mathbf{Y},i}} \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XY}}\mathbf{w}_{\mathbf{Y},i} \\ &\quad \text{s.t. } \|\mathbf{X}\mathbf{w}_{\mathbf{X},i}\| = \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i} = 1 \\ &\quad \|\mathbf{Y}\mathbf{w}_{\mathbf{Y},i}\| = \mathbf{w}'_{\mathbf{Y},i}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} = 1 \\ &\quad \text{and for } j < i : \begin{array}{l}\mathbf{w}'_{\mathbf{X},j}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},i} = 0, \\ \mathbf{w}'_{\mathbf{Y},j}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},i} = 0.\end{array}\end{aligned}$$

- The CCA problem can be reformulated as an ordinary eigenvalue problem:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XX}}^{-1}\mathbf{S_{XY}} \\ \mathbf{S_{YY}}^{-1}\mathbf{S_{YX}} & \mathbf{0} \end{pmatrix}\begin{pmatrix}\mathbf{w_X} \\ \mathbf{w_Y}\end{pmatrix} = \lambda\begin{pmatrix}\mathbf{w_X} \\ \mathbf{w_Y}\end{pmatrix}.$$

  This eigenvalue problem can be made symmetric by introducing $\mathbf{v_X} = \mathbf{S_{XX}}^{1/2}\mathbf{w_X}$ and $\mathbf{v_Y} = \mathbf{S_{YY}}^{1/2}\mathbf{w_Y}$:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XX}}^{-1/2}\mathbf{S_{XY}}\mathbf{S_{YY}}^{-1/2} \\ \mathbf{S_{YY}}^{-1/2}\mathbf{S_{YX}}\mathbf{S_{XX}}^{-1/2} & \mathbf{0} \end{pmatrix}\begin{pmatrix}\mathbf{v_X} \\ \mathbf{v_Y}\end{pmatrix} = \lambda\begin{pmatrix}\mathbf{v_X} \\ \mathbf{v_Y}\end{pmatrix}.$$

  Note that this eigenvalue problem is of the form of Eq. (2), so here $\mathbf{v_X}$ and $\mathbf{v_Y}$ are the left and right singular vectors of $\mathbf{S_{XX}}^{-1/2}\mathbf{S_{XY}}\mathbf{S_{YY}}^{-1/2}$. The weight vectors can be retrieved as $\mathbf{w_X} = \mathbf{S_{XX}}^{-1/2}\mathbf{v_X}$ and $\mathbf{w_Y} = \mathbf{S_{YY}}^{-1/2}\mathbf{v_Y}$.

  By the orthogonality of the singular vectors, we can derive in an alternative way that projections onto noncorresponding canonical directions are uncorrelated: $0 = \mathbf{v}'_{\mathbf{X},i}\mathbf{v}_{\mathbf{X},j} = \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XX}}\mathbf{w}_{\mathbf{X},j}$, and $0 = \mathbf{v}'_{\mathbf{Y},i}\mathbf{v}_{\mathbf{Y},j} = \mathbf{w}'_{\mathbf{Y},i}\mathbf{S_{YY}}\mathbf{w}_{\mathbf{Y},j}$. Also, we find that $0 = \mathbf{v}'_{\mathbf{X},i}\mathbf{S_{XX}}^{-1/2}\mathbf{S_{XY}}\mathbf{S_{YY}}^{-1/2}\mathbf{v}_{\mathbf{Y},j} = \mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XY}}\mathbf{w}_{\mathbf{Y},j}$.

- As a last remark, we note that CCA where one of both data spaces is one-dimensional is equivalent to least squares regression (LSR).

4.2.4 Dual

To derive the dual, again note that the (minimum norm[3]) $\mathbf{w_X}$ and $\mathbf{w_Y}$ will lie in the column space of $\mathbf{X}$ and $\mathbf{Y}$, respectively (thus, analogously to Eq. (3),

[3] The motivation for taking the minimum norm solution is as follows: first of all, we need to make a choice in cases where there is an indeterminacy as is when the rows of $\mathbf{X}$ and/or $\mathbf{Y}$ do not span the whole space. And a component of the weight vectors orthogonal to the data would never contribute to the correlation of a projection of the data onto this weight vector anyway; the projection onto this orthogonal direction would be zero. We do not get any information concerning the orthogonal subspace, and thus do not want $\mathbf{w}$ to make any unmotivated predictions on this. In this chapter we always look for minimum norm solutions.

$\mathbf{w_X} = \mathbf{X}'\boldsymbol{\alpha}_\mathbf{X}$ and $\mathbf{w_Y} = \mathbf{Y}'\boldsymbol{\alpha}_\mathbf{Y}$; see also [3] for a more detailed explanation). Thus we can write

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_\mathbf{X} \\ \mathbf{Y}'\boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} \end{pmatrix} \begin{pmatrix} \mathbf{X}'\boldsymbol{\alpha}_\mathbf{X} \\ \mathbf{Y}'\boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix}$$

$$\Downarrow \text{ multiplying left with } \begin{pmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Y} \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{0} & \mathbf{X S_{XY} Y}' \\ \mathbf{Y S_{YX} X}' & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{X S_{XX} X}' & \mathbf{0} \\ \mathbf{0} & \mathbf{Y S_{YY} Y}' \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X^2} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y^2} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix}.$$

Projections of test points $\mathbf{x}_{\text{test}}$ and $\mathbf{y}_{\text{test}}$ onto the CCA directions corresponding to $\boldsymbol{\alpha}_\mathbf{X}$ and $\boldsymbol{\alpha}_\mathbf{Y}$ can then be carried out as

$$\sum_{i=1}^{n} \alpha_{\mathbf{X},i} k(\mathbf{x}_i, \mathbf{x}_{\text{test}}), \text{ and } \sum_{i=1}^{n} \alpha_{\mathbf{Y},i} k(\mathbf{y}_i, \mathbf{y}_{\text{test}}). \tag{11}$$

4.2.5 Regularization

Primal problem

Regularization is often necessary in doing CCA for the following reason. The scatter matrices $\mathbf{S_{XX}}$ and $\mathbf{S_{YY}}$ are proportional to finite sample estimates of the covariance matrices. This generally leads to poor performance in case of small eigenvalues of these covariances. Remember the generalized eigenvalue problem is (theoretically) equivalent with a standard eigenvalue problem where the right-hand side matrix containing the scatter matrices is inverted. Any fluctuation of the smallest eigenvalue will thus be blown up in the inverse. To counteract this effect, one often adds a diagonal to the scatter matrices, or equivalently to each of their eigenvalues [3]. In this way, a bias is introduced, but it is hoped that for a certain bias, the total variance will be lower than the case when no bias is present.

An equivalent way to view this is, as presented above in the ridge regression derivation, by interpreting the regularization as a reduction of the effective number of degrees of freedom. Generalization will be more likely to be good.

The primal regularized problem is thus

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S_{XX}} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S_{YY}} + \gamma\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}.$$

Intuitively, this type of regularization boils down to trusting correlations along high-variance directions more than along low-variance directions. Or, equivalently, it corresponds to a modified optimization problem where the constraints contain an additional term constraining the norm of $\mathbf{w_X}$ and $\mathbf{w_Y}$, similarly to the ridge regression cost function.

Note that RCCA with one of both spaces one-dimensional is equivalent to ridge regression (RR).

Dual problem

The dual of this generalized eigenvalue problem can be derived in the same way as the unregularized problem, leading to:

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X^2} + \gamma \mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y^2} + \gamma \mathbf{K_Y} \end{pmatrix} \begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix}. \quad (12)$$

In the dual case, the need for regularization is often even stronger than in the primal case. This is because the feature space is often infinite-dimensional, so that the freedom to find correlations is much too high. All correlations would be equal to 1, which means no generalization is possible at all. Penalizing a large weight vector as above thus makes sense to improve generalization.

When both the kernels have full rank, left-multiplication on both sides of Eq. (12) with $\begin{pmatrix} \mathbf{K_X^{-1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y^{-1}} \end{pmatrix}$ reveals that this generalized eigenvalue problem is equivalent with

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_Y} \\ \mathbf{K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X} + \gamma \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y} + \gamma \mathbf{I} \end{pmatrix} \begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix}. \quad (13)$$

Kernel matrices are often rank deficient, however (e.g. when they are centered). In that case the solutions of Eq. (13) are still solutions for Eq. (12), but the converse is no longer always true. The reason is that for any generalized eigenvector $\begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix}$ of Eq. (13) and thus of Eq. (12), $\begin{pmatrix} \alpha_\mathbf{X} + \alpha_{\mathbf{X}0} \\ \alpha_\mathbf{Y} + \alpha_{\mathbf{Y}0} \end{pmatrix}$, where $\alpha_{\mathbf{X}0}$ and $\alpha_{\mathbf{Y}0}$ are arbitrary vectors lying respectively in the null spaces of $\mathbf{K_X}$ and $\mathbf{K_Y}$, is also an eigenvector with the same eigenvalue of Eq. (12) but generally not of Eq. (13). However, similarly as in the ridge regression derivation, it can be seen that these components $\alpha_{\mathbf{X}0}$ and $\alpha_{\mathbf{Y}0}$ play no role in the calculation of Eq. (11). This is because the weight vectors $\mathbf{w_X} = \mathbf{X}'(\alpha_\mathbf{X} + \alpha_{\mathbf{X}0}) = \mathbf{X}'\alpha_\mathbf{X}$ and $\mathbf{w_Y} = \mathbf{Y}'(\alpha_\mathbf{Y} + \alpha_{\mathbf{Y}0}) = \mathbf{Y}'\alpha_\mathbf{Y}$ are unaffected by the components in the null spaces of $\mathbf{K_X}$ and $\mathbf{K_Y}$. Therefore, we can choose to solve either Eq. (12) or Eq. (13).

4.3 Partial Least Squares

Partial least squares (PLS, introduced in [33, 34]; see also [14] for a good review) can be interpreted in two ways. The first PLS component is the maximally regularized version of the first CCA component (the case where $\gamma \to \infty$, after rescaling the eigenvalues by multiplying them with γ). Another view is as a covariance maximizer instead of a correlation maximizer, this again for the first PLS component. Whereas all PLS formulations compute the first component in the same way, there is no one way to compute the other components. We will present two variants: so-called EZ-PLS, which consists of only one eigenvalue decomposition (or a singular value decomposition) and which is used mainly for exploratory purposes (similar to CCA), and regression-PLS which is a more involved version that is most widely used in (multivariate) regression applications.

Because of the iterative way PLS components are computed in, and because of the fact that there exist several variants of PLS, the discussion is somewhat more involved. We will first give a general discussion on the cost function optimized in all PLS formulations, followed by the eigenproblem optimizing this cost function. Next, we will shortly go into some computational aspects. Finally, we will show the particularities of the two PLS formulations EZ-PLS and regression-PLS, followed by a discussion of the regression step in regression-PLS. Again, a primal and a dual (see [19] where this was first derived) formulation will be provided.

4.3.1 Cost Function

Maximize the sample *covariance*[4] between a projection of $\mathbf{X}$ and a projection of $\mathbf{Y}$:

$$\begin{aligned}\{\mathbf{w_X}, \mathbf{w_Y}\} &= \mathrm{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \frac{(\mathbf{X}\mathbf{w_X})'(\mathbf{Y}\mathbf{w_Y})}{\sqrt{\mathbf{w}'_\mathbf{X}\mathbf{w_X}}\sqrt{\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}}},\\ &= \mathrm{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \frac{\mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y}}{\sqrt{\mathbf{w}'_\mathbf{X}\mathbf{w_X}}\sqrt{\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}}}.\end{aligned}$$

This is equivalent to maximizing the sample covariance, or the ‘fit’ subject to constraints:

$$\begin{aligned}\{\mathbf{w_X}, \mathbf{w_Y}\} &= \mathrm{argmax}_{\mathbf{w_X},\mathbf{w_Y}} \mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y}\\ &\quad \text{s.t. } \|\mathbf{w_X}\|^2 = \mathbf{w}'_\mathbf{X}\mathbf{w_X} = 1, \|\mathbf{w_Y}\|^2 = \mathbf{w}'_\mathbf{Y}\mathbf{w_Y} = 1,\end{aligned}$$

and equivalent to minimizing the misfit subject to these constraints:

$$\begin{aligned}\{\mathbf{w_X}, \mathbf{w_Y}\} &= \mathrm{argmin}_{\mathbf{w_X},\mathbf{w_Y}} \|\mathbf{X}\mathbf{w_X} - \mathbf{Y}\mathbf{w_Y}\|^2\\ &\quad \text{s.t. } \|\mathbf{w_X}\|^2 = 1, \|\mathbf{w_Y}\|^2 = 1.\end{aligned}$$

4.3.2 Primal

We solve the second formulation of the problem. Differentiating the Lagrangian $\mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = \mathbf{w}'_\mathbf{X}\mathbf{S_{XY}}\mathbf{w_Y} - \lambda_\mathbf{X}\mathbf{w}'_\mathbf{X}\mathbf{w_X} - \lambda_\mathbf{Y}\mathbf{w}'_\mathbf{Y}\mathbf{w_Y}$ with respect to $\mathbf{w_X}$ and $\mathbf{w_Y}$ and equating to 0 gives

[4] Note the difference between CCA where *correlation* was maximized.

$$\begin{cases} \frac{\partial}{\partial \mathbf{w_X}} \mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = 0, \\ \frac{\partial}{\partial \mathbf{w_Y}} \mathcal{L}(\mathbf{w_X}, \mathbf{w_Y}, \lambda_\mathbf{X}, \lambda_\mathbf{Y}) = 0, \end{cases}$$

$$\Rightarrow \begin{cases} \mathbf{S_{XY}} \mathbf{w_Y} = \lambda_\mathbf{X} \mathbf{w_X}. \\ \mathbf{S_{YX}} \mathbf{w_X} = \lambda_\mathbf{Y} \mathbf{w_Y}. \end{cases}$$

Since from this

$$\lambda_\mathbf{X} \mathbf{w}'_\mathbf{X} \mathbf{w_X} = \mathbf{w}'_\mathbf{X} \mathbf{S_{XY}} \mathbf{w_Y} = \mathbf{w}'_\mathbf{Y} \mathbf{S_{YX}} \mathbf{w_X} = \lambda_\mathbf{Y} \mathbf{w}'_\mathbf{Y} \mathbf{w_Y},$$

and since $\mathbf{w}'_\mathbf{X} \mathbf{w_X} = \mathbf{w}'_\mathbf{Y} \mathbf{w_Y} = 1$, we find that $\lambda_\mathbf{X} = \lambda_\mathbf{Y} = \lambda$. Thus

$$\begin{cases} \mathbf{S_{XY}} \mathbf{w_Y} = \lambda \mathbf{w_X}, \\ \mathbf{S_{YX}} \mathbf{w_X} = \lambda \mathbf{w_Y}. \end{cases} \tag{14}$$

Or, stated in another way as an eigenvalue problem,

$$\begin{pmatrix} \mathbf{0} & \mathbf{S_{XY}} \\ \mathbf{S_{YX}} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{w_X} \\ \mathbf{w_Y} \end{pmatrix}. \tag{15}$$

This eigenvalue problem has d eigenvalues, corresponding to a covariance between projections onto $\mathbf{w_X}$ and $\mathbf{w_Y}$. The largest one with its eigenvector corresponds to the optimum of the cost function described earlier.

Note that Eq. (15) is of the form of Eq. (2). Thus the EZ-PLS problem can be solved by calculating the singular value decomposition of $\mathbf{S_{XY}}$.

4.3.3 Dual

The dual problem can easily be found by using Eq. (3):

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix},$$

which includes all solutions of

$$\begin{pmatrix} \mathbf{0} & \mathbf{K_Y} \\ \mathbf{K_X} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} = \lambda \begin{pmatrix} \boldsymbol{\alpha}_\mathbf{X} \\ \boldsymbol{\alpha}_\mathbf{Y} \end{pmatrix} \tag{16}$$

as its solutions as well. Similarly, as in CCA this is the formulation of the dual problem that is solved, since it does not suffer from indeterminacies.

Projections of test points $\mathbf{x}_{\text{test}}$ and $\mathbf{y}_{\text{test}}$ onto the PLS directions corresponding to $\boldsymbol{\alpha}_X$ and $\boldsymbol{\alpha}_Y$ can then be computed as

$$\sum_{i=1}^{n} \alpha_{\mathbf{X},i} k(\mathbf{x}_i, \mathbf{x}_{\text{test}}), \text{ and } \sum_{i=1}^{n} \alpha_{\mathbf{Y},i} k(\mathbf{y}_i, \mathbf{y}_{\text{test}}).$$

It is important to note that the first component corresponds to maximally regularized RCCA. Taking more than one component lessens this regularization in an alternative way in comparison to RCCA. This will be the subject of the remainder of this section on PLS.

4.3.4 Nonlinear Iterative Partial Least Squares and Primal–Dual Symmetry in PLS

A straightforward way to solve for the largest eigenvector of Eq. (15) could be by using the power method. However, thanks to the structure of the eigenvalue problems at hand, it can be solved by using the so-called nonlinear iterative partial least squares (NIPALS) method [33]. Note that, from Eqs. (15) and (16):

- $\mathbf{YY}'\mathbf{XX}'\alpha_{\mathbf{X}} = \lambda^2\alpha_{\mathbf{X}}$.
- $\mathbf{X}'\mathbf{YY}'\mathbf{X}\mathbf{w}_{\mathbf{X}} = \lambda^2\mathbf{w}_{\mathbf{X}}$.
- $\mathbf{XX}'\mathbf{YY}'\alpha_{\mathbf{Y}} = \lambda^2\alpha_{\mathbf{Y}}$.
- $\mathbf{Y}'\mathbf{XX}'\mathbf{Y}\mathbf{w}_{\mathbf{Y}} = \lambda^2\mathbf{w}_{\mathbf{Y}}$.

Thus it follows that both the primal and the dual eigenvalue problem are actually solved at the same time, using the following ‘power’ method:

0. Fix initial value $\mathbf{w}_{\mathbf{Y}}$, normalize. Then iterate over steps 1–4.
1. $\alpha_{\mathbf{X}} = \mathbf{Y}\mathbf{w}_{\mathbf{Y}}$.
2. $\mathbf{w}_{\mathbf{X}} = \mathbf{X}'\alpha_{\mathbf{X}}$, normalize $\mathbf{w}_{\mathbf{X}}$ to unit length.
3. $\alpha_{\mathbf{Y}} = \mathbf{X}\mathbf{w}_{\mathbf{X}}$.
4. $\mathbf{w}_{\mathbf{Y}} = \mathbf{Y}'\alpha_{\mathbf{Y}}$, normalize $\mathbf{w}_{\mathbf{Y}}$ to unit length.

After convergence, the normalizations carried out in steps 2 and 4 both amount to a division by λ; then $\mathbf{w}_{\mathbf{X}} = \frac{1}{\lambda}\mathbf{X}'\alpha_{\mathbf{X}}$ and $\mathbf{w}_{\mathbf{Y}} = \frac{1}{\lambda}\mathbf{Y}'\alpha_{\mathbf{Y}}$.

In case the feature vectors $\mathbf{X}$ are only implicitly determined by a kernel function, steps 2 and 3 must be combined in one step:

2,3. $\alpha_{\mathbf{Y}} = \mathbf{K}_{\mathbf{X}}\alpha_{\mathbf{X}}$, normalize.

It can be seen that each of these weight vectors or dual vectors converge to the eigenvector of the above four eigenvalue problems (combining four steps following each other gives the power method for one of these four eigenvalue problems). Since these are equivalent with Eqs. (15) and (16), they converge to the PLS weight vectors and dual vectors.

In this way, we can solve efficiently for the largest singular value and singular vectors. Only this one component is not enough to solve most practical problems, however. We discuss two ways to extract more information present in the data: what we call EZ-PLS and regression-PLS. For both methods first the primal versions will be discussed, then afterwards the dual.

4.3.5 EZ-PLS

Primal

In EZ-PLS, the other PLS directions are the other eigenvectors corresponding to a different covariance (eigenvalue) λ. This can be accomplished by using an iterative deflation scheme:

1. Initialize: ${\mathbf{S_{XY}}}^0 \leftarrow \mathbf{S_{XY}}$.
2. Compute the largest singular value of ${\mathbf{S_{XY}}}^i$ with NIPALS. This gives the ith PLS component. Normalize so that $\|\mathbf{w}_{\mathbf{X},i}\| = \|\mathbf{w}_{\mathbf{Y},i}\| = 1$.
3. Deflate the scatter matrices:

$$ {\mathbf{S_{XY}}}^{i+1} \leftarrow {\mathbf{S_{XY}}}^i - \lambda_i \mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{Y},i}. $$

   The rank of ${\mathbf{S_{XY}}}^{i+1}$ is 1 less than the rank of ${\mathbf{S_{XY}}}^i$.
4. When the number of desired components (necessarily lower than the rank of $\mathbf{S_{XY}}$) is not yet reached, go to step 2.

The deflation of the $\mathbf{X}^i$ matrix for EZ-PLS, in order to get the desired deflation of the cross-scatter matrix, is

$$ \mathbf{X}^{i+1} \leftarrow \mathbf{X}^i - \mathbf{X}^i \mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}. $$

Similarly, one could do the deflation of the $\mathbf{Y}^i$ matrix

$$ \mathbf{Y}^{i+1} \leftarrow \mathbf{Y}^i - \mathbf{Y}^i \mathbf{w}_{\mathbf{Y},i}\mathbf{w}'_{\mathbf{Y},i}, $$

also leading to the same desired deflation of the cross-scatter matrix.

Dual

Taking Eq. (3) or equivalently the NIPALS iteration into account, the deflation of the kernel matrices corresponding to the EZ-PLS deflation is found to be

$$ \mathbf{K}_{\mathbf{X}}^{i+1} \leftarrow \mathbf{K}_{\mathbf{X}}^i - \frac{1}{\lambda_i^2}\mathbf{K}_{\mathbf{X}}^i \boldsymbol{\alpha}_{\mathbf{X},i}\boldsymbol{\alpha}'_{\mathbf{X},i}\mathbf{K}_{\mathbf{X}}^i = \mathbf{K}_{\mathbf{X}}^i - \boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}. $$

Properties

- Since the $\mathbf{w}_{\mathbf{X},i}$ and the $\mathbf{w}_{\mathbf{Y},i}$ are the left and right singular vectors of $\mathbf{S_{XY}}$, all $\mathbf{w}_{\mathbf{X},i}$ are orthogonal to each other, and all $\mathbf{w}_{\mathbf{Y},i}$ are orthogonal to each other.
- For the same reason, if $i \neq j$: $\mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XY}}\mathbf{w}_{\mathbf{Y},j} = 0$. In other words, projections onto noncorresponding $\mathbf{w}_{\mathbf{X},i}$ and $\mathbf{w}_{\mathbf{Y},j}$ are uncorrelated.
- All EZ-PLS components can be calculated at once by optimizing the same cost function as for the first component, taking the first (orthogonality) property into account as an additional constraint.

The EZ-PLS form is the easiest, in the sense that because of the nature of the deflation, it is in fact not more than solving for the most important singular vectors of $\mathbf{S_{XY}}$. That is why it is discussed here; it is less useful in practice.

4.3.6 Regression-PLS

Whereas EZ-PLS is not often used for regression (note that it is entirely symmetric between $\mathbf{X}$ and $\mathbf{Y}$, whereas regression is not; it is rather used for modelling though), regression-PLS is the PLS formulation that is generally preferred for (multivariate) regression (see [14]). We will first discuss the deflations that are characteristic for regression-PLS. Further on we will explain how regression can be carried out using the results from these deflations.

Primal

The difference between EZ-PLS and Regression-PLS lies in the way the deflation is carried out. Regression-PLS has the intention of modelling one (possibly) vectorial variable $\mathbf{Y}$ with the other vectorial variable $\mathbf{X}$, hence the name.[5] It is thus asymmetric between the two spaces, which is expressed in the deflation step:

2,4. Deflate by orthogonalizing $\mathbf{X}^i$ to its projection onto the weight vector $\mathbf{w}_{\mathbf{X},i}$, $\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}$, and recomputing the scatter matrix:

$$\mathbf{X}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\right)\mathbf{X}^i = \mathbf{X}^i - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\mathbf{X}^i, \quad (17)$$

$$= \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{X}^i. \quad (18)$$

Finally (see later, Eq. (28)) we will perform a regression of $\mathbf{Y}$ based on the $\boldsymbol{\alpha}_{\mathbf{Y},i}$. (The $\boldsymbol{\alpha}_{\mathbf{Y},i}$ can be computed from $\mathbf{X}$ as will become clear later, see Eq. (27).) Therefore, we also deflate $\mathbf{Y}^i$ with $\boldsymbol{\alpha}_{\mathbf{Y},i}$ to remove the information captured by the ith iteration:

$$\mathbf{Y}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\right)\mathbf{Y}^i. \quad (19)$$

This boils down to the following deflation of the scatter matrix:

$$\mathbf{S_{XY}}^{i+1} \leftarrow \mathbf{S_{XY}}^i - \frac{\lambda_i}{\mathbf{w}'_{\mathbf{X},i}\mathbf{S_{XX}}^i\mathbf{w}_{\mathbf{X},i}}\mathbf{S_{XX}}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{Y},i}.$$

The philosophy behind this kind of deflation is as follows: after step i, part of the information in $\mathbf{X}^i$, namely its projection $\boldsymbol{\alpha}_{\mathbf{Y},i}$ onto the ith PLS direction $\mathbf{w}_{\mathbf{X},i}$, is captured already: the component $\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{X}^i$ of $\mathbf{X}^i$ (along $\boldsymbol{\alpha}_{\mathbf{Y},i}$) perfectly models the component $\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{Y}^i$ of $\mathbf{Y}^i$. This information should not be used or modelled again in next steps, so it is 'subtracted' from both $\mathbf{X}^i$ and $\mathbf{Y}^i$. In the next step, the direction of maximal covariance between the remaining information $\mathbf{X}^{i+1}$ and $\mathbf{Y}^{i+1}$ is found, and so on.

[5] In literature this form of PLS is best known as PLS2, or PLS1 for the case where $\mathbf{Y}$ is one-dimensional.

Dual

Using Eqs. (18) and (19), the deflation of the kernel matrices corresponding to the regression-PLS deflation can be shown to be

$$\mathbf{K}_{\mathbf{X}}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\mathbf{K}_{\mathbf{X}}^{i}\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right).$$

Analogously,

$$\mathbf{K}_{\mathbf{Y}}^{i+1} \leftarrow \left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\mathbf{K}_{\mathbf{Y}}^{i}\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right).$$

Properties

- The different weight vectors $\mathbf{w}_{\mathbf{Y},i}$ are *not* orthogonal (it is even possible that they are all collinear, e.g. in the case where $\mathbf{Y}$ is one-dimensional). The different weight vectors $\mathbf{w}_{\mathbf{X},i}$, however, are orthogonal. Using Eq. (17),

$$\mathbf{w}'_{\mathbf{X},i}\mathbf{S}_{\mathbf{XY}}^{i+1} = \mathbf{w}'_{\mathbf{X},i}\left(\left(\mathbf{I} - \frac{\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}}{\mathbf{w}'_{\mathbf{X},i}\mathbf{X}^{i\prime}\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}}\right)\mathbf{X}^i\right)'\mathbf{Y}^{i+1} = \mathbf{0},$$

so that $\mathbf{w}_{\mathbf{X},i}$ is in the left null space of ${\mathbf{S}_{\mathbf{XY}}}^{i+1}$. Since $\mathbf{w}_{\mathbf{X},i+1}$ is a left singular vector of ${\mathbf{S}_{\mathbf{XY}}}^{i+1}$ this means that $\mathbf{w}_{\mathbf{X},i+1}$ will be orthogonal to $\mathbf{w}_{\mathbf{X},i}$. By replacing the left-most $\mathbf{X}^i$ in the above equation by $\left(\mathbf{I} - \frac{\mathbf{X}^{i-1}\mathbf{w}_{\mathbf{X},i-1}\mathbf{w}'_{\mathbf{X},i-1}\mathbf{X}^{i-1\prime}}{\mathbf{w}'_{\mathbf{X},i-1}\mathbf{X}^{i-1\prime}\mathbf{X}^{i-1}\mathbf{w}_{\mathbf{X},i-1}}\right)\mathbf{X}^{i-1}$, and so on for $\mathbf{X}^{i-1},\ldots$, one can see that also for $j < i$, $\mathbf{w}_{\mathbf{X},j}$ is orthogonal to $\mathbf{w}_{\mathbf{X},i}$. Thus, all $\mathbf{w}_{\mathbf{X},i}$ are mutually orthogonal:

$$\mathbf{W}'_{\mathbf{X}}\mathbf{W}_{\mathbf{X}} = \mathbf{I},$$

where $\mathbf{W}_{\mathbf{X}}$ represents the matrix built by stacking the vectors $\mathbf{w}_{\mathbf{X},i}$ next to each other.

- The vectors $\alpha_{\mathbf{Y},i}$ are mutually orthogonal. Using Eq. (18), for $i \leq j$ one has:

$$\mathbf{X}^{j\prime}\alpha_{\mathbf{Y},i} = \mathbf{X}^{i\prime}\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},j-1}\alpha'_{\mathbf{Y},j-1}}{\alpha'_{\mathbf{Y},j-1}\alpha_{\mathbf{Y},j-1}}\right)\alpha_{\mathbf{Y},i}.$$

For $j = i + 1$, this is immediately proven to be zero. When this product is zero for all $j : i < j < j*$, $\alpha'_{\mathbf{Y},j}\alpha_{\mathbf{Y},i} = \mathbf{w}'_{\mathbf{X},j}\mathbf{X}^{j\prime}\alpha_{\mathbf{Y},i} = 0$, and the matrices between brackets in the above product commute. Since this is indeed true for $j = i + 1$, by induction it is proved for all $i < j$ that:

$$\mathbf{X}^{j\prime}\alpha_{\mathbf{Y},i} = \mathbf{0}, \tag{20}$$

and thus by left multiplication with $\mathbf{w}_{\mathbf{X},j}$

$$\alpha'_{\mathbf{Y},j}\alpha_{\mathbf{Y},i} = 0. \tag{21}$$

Note that since $\alpha_{\mathbf{Y},i} = \mathbf{X}^i\mathbf{w}_{\mathbf{X},i}$, this means that the projections $\alpha_{\mathbf{Y},i}$ of $\mathbf{X}^i$ onto their weight vectors $\mathbf{w}_{\mathbf{X},i}$ are uncorrelated with each other. This property may remind you of CCA.

- This orthogonality property in Eq. (21) of the $\alpha_{\mathbf{Y},i}$ leads to the fact that

$$\begin{aligned}\mathbf{w}_{\mathbf{Y},i} = \mathbf{Y}^{i'}\alpha_{\mathbf{Y},i} = \mathbf{Y}'\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},1}\alpha'_{\mathbf{Y},1}}{\alpha'_{\mathbf{Y},1}\alpha_{\mathbf{Y},1}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i-1}\alpha'_{\mathbf{Y},i-1}}{\alpha'_{\mathbf{Y},i-1}\alpha_{\mathbf{Y},i-1}}\right)\alpha_{\mathbf{Y},i} \\ \Rightarrow \mathbf{w}_{\mathbf{Y},i} = \mathbf{Y}'\alpha_{\mathbf{Y},i},\end{aligned} \tag{22}$$

up to a normalization.

- Furthermore, one finds that for $i < j$:

$$\begin{aligned}\mathbf{X}^j\mathbf{w}_{\mathbf{X},i} &= \left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},j-1}\alpha'_{\mathbf{Y},j-1}}{\alpha'_{\mathbf{Y},j-1}\alpha_{\mathbf{Y},j-1}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\mathbf{X}^i\mathbf{w}_{\mathbf{X},i}, \\ &= \left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},j-1}\alpha'_{\mathbf{Y},j-1}}{\alpha'_{\mathbf{Y},j-1}\alpha_{\mathbf{Y},j-1}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\alpha_{\mathbf{Y},i}, \\ &= \mathbf{0}.\end{aligned} \tag{23}$$

This generally does not hold for $i \geq j$.

- Another consequence of Eq. (21) is, for $i < j$:

$$\begin{aligned}\mathbf{Y}^{j'}\alpha_{\mathbf{Y},i} &= \mathbf{Y}^{i'}\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},j-1}\alpha'_{\mathbf{Y},j-1}}{\alpha'_{\mathbf{Y},j-1}\alpha_{\mathbf{Y},j-1}}\right)\alpha_{\mathbf{Y},i}, \\ &= \mathbf{0}.\end{aligned} \tag{24}$$

- And thus also, for $i < j$:

$$\begin{aligned}\alpha'_{\mathbf{X},j}\alpha_{\mathbf{Y},i} &= \mathbf{w}_{\mathbf{X},j}\mathbf{Y}^{j'}\alpha_{\mathbf{Y},i}, \\ &= 0.\end{aligned} \tag{25}$$

- From this it follows that

$$\begin{aligned}\mathbf{w}_{\mathbf{X},i} = \mathbf{X}^{i'}\alpha_{\mathbf{X},i} = \mathbf{X}'\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},1}\alpha'_{\mathbf{Y},1}}{\alpha'_{\mathbf{Y},1}\alpha_{\mathbf{Y},1}}\right)\ldots\left(\mathbf{I} - \frac{\alpha_{\mathbf{Y},i-1}\alpha'_{\mathbf{Y},i-1}}{\alpha'_{\mathbf{Y},i-1}\alpha_{\mathbf{Y},i-1}}\right)\alpha_{\mathbf{X},i} \\ \Rightarrow \mathbf{w}_{\mathbf{X},i} = \mathbf{X}'\alpha_{\mathbf{X},i},\end{aligned} \tag{26}$$

up to a normalization factor.

Thus as a summary:

$$\begin{aligned}
\mathbf{w}_{\mathbf{X},i} &\propto \mathbf{X}'\boldsymbol{\alpha}_{\mathbf{X},i},\\
\mathbf{w}_{\mathbf{Y},i} &\propto \mathbf{Y}'\boldsymbol{\alpha}_{\mathbf{Y},i},\\
\mathbf{w}'_{\mathbf{X},j}\mathbf{w}_{\mathbf{X},i} &= 0,\\
\boldsymbol{\alpha}'_{\mathbf{Y},j}\boldsymbol{\alpha}_{\mathbf{Y},i} &= 0,\\
\boldsymbol{\alpha}'_{\mathbf{X},j}\boldsymbol{\alpha}_{\mathbf{Y},i} &= 0 \text{ for } i < j,\\
\mathbf{X}^j\mathbf{w}_{\mathbf{X},i} &= \mathbf{0} \text{ for } i < j,\\
\mathbf{Y}^{j\prime}\boldsymbol{\alpha}_{\mathbf{Y},i} &= \mathbf{0} \text{ for } i < j.
\end{aligned}$$

4.3.7 Final Regression in Regression-PLS

Primal

The entire regression-PLS algorithm is composed of a (generally noninvertible) linear mapping of $\mathbf{X}$ towards k so-called *latent variables* (in the current context we would rather call them dual variables) $\boldsymbol{\alpha}_{\mathbf{Y},i} = \mathbf{X}^i\mathbf{w}_{\mathbf{X},i}$, followed by a regression of $\mathbf{Y}$ on $\mathbf{A}_{\mathbf{Y}}$, where $\mathbf{A}_{\mathbf{Y}}$ contains $\boldsymbol{\alpha}_{\mathbf{Y},i}$ as its columns.

The part of $\mathbf{X}$ that has been deflated and thus will be used for regression is equal to the sum $\sum_{i=1}^{k} \frac{\boldsymbol{\alpha}_{\mathbf{Y},i}\boldsymbol{\alpha}'_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}\mathbf{X}^i = \mathbf{A}_{\mathbf{Y}}\mathbf{P}'$, where the vectors $\mathbf{p}_i = \mathbf{X}^{i\prime}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}$ make up the columns of $\mathbf{P}$. Analogously, define $\mathbf{c}_i = \mathbf{Y}^{i\prime}\frac{\boldsymbol{\alpha}_{\mathbf{Y},i}}{\boldsymbol{\alpha}'_{\mathbf{Y},i}\boldsymbol{\alpha}_{\mathbf{Y},i}}$ making up the columns of $\mathbf{C}$.

Now, if we go on with the deflations until the rank of $\mathbf{X}^i$ is zero,[6] the space spanned by the orthogonal vectors $\boldsymbol{\alpha}_{\mathbf{Y},i}$ is complete and we have that

$$\mathbf{X} = \mathbf{A}_{\mathbf{Y}}^{\text{tot}}\mathbf{P}^{\text{tot}\prime} = \mathbf{A}_{\mathbf{Y}}\mathbf{P}' + \mathbf{A}_{\mathbf{Y}}^{\text{rem}}\mathbf{P}^{\text{rem}\prime} = \mathbf{A}_{\mathbf{Y}}\mathbf{P}' + \mathbf{E}_{\mathbf{X}},$$

with $\mathbf{E}_{\mathbf{X}}$ the part of $\mathbf{X}$ that is not used in regression when the components corresponding to $\mathbf{A}_{\mathbf{Y}}^{\text{rem}}$ are not kept. Also, because of Eq. (23) and the definition of $\mathbf{P}$: $\mathbf{p}_j{}'\mathbf{w}_{\mathbf{X},i} = 0$ for $i < j$, and thus:

$$\mathbf{P}^{\text{rem}\prime}\mathbf{W}_{\mathbf{X}} = \mathbf{0}.$$

This leads to the linear mapping from $\mathbf{X}$ to $\mathbf{A}_{\mathbf{Y}}$:

$$\begin{aligned}
&\mathbf{A}_{\mathbf{Y}}\mathbf{P}'\mathbf{W}_{\mathbf{X}} = \mathbf{X}\mathbf{W}_{\mathbf{X}}\\
&\quad\Rightarrow \mathbf{A}_{\mathbf{Y}} = \mathbf{X}\mathbf{W}_{\mathbf{X}}\left(\mathbf{P}'\mathbf{W}_{\mathbf{X}}\right)^{-1},
\end{aligned} \tag{27}$$

where the matrix to be inverted is lower triangular (again because $\mathbf{p}_j{}'\mathbf{w}_{\mathbf{X},i} = 0$ for $i < j$), so the inversion can be carried out efficiently.

The regression from the latent variables $\boldsymbol{\alpha}_{\mathbf{Y}}$ towards $\mathbf{Y}$ is given by

[6] Note that the number of deflations k will always be smaller (or equal, in full LSR) than the rank of $\mathbf{X}$. This results in matrices $\mathbf{W}_{\mathbf{X}}, \mathbf{W}_{\mathbf{Y}}, \mathbf{A}'_{\mathbf{X}}, \mathbf{A}'_{\mathbf{Y}}, \mathbf{P}$, and $\mathbf{C}$ all having k columns.

$$\mathbf{Y} = \sum_{i=1}^{k} \frac{\alpha_{\mathbf{Y},i}\alpha'_{\mathbf{Y},i}}{\alpha'_{\mathbf{Y},i}\alpha_{\mathbf{Y},i}} \mathbf{Y}^i + \mathbf{Y}^{k+1} = \mathbf{A}_\mathbf{Y}\mathbf{C}' + \mathbf{E}_\mathbf{Y}, \tag{28}$$

where $\mathbf{E}_\mathbf{Y} = \mathbf{Y}^{k+1}$ is the part of $\mathbf{Y}$ that is not predicted by the first k PLS components (the misfit).

Thus, the entire PLS regression formula is given by

$$\mathbf{y}_{\text{pred}} = \left[\mathbf{W}_\mathbf{X}\left(\mathbf{P}'\mathbf{W}_\mathbf{X}\right)^{-1}\mathbf{C}'\right]' \mathbf{x}_{\text{pred}} = \left[\mathbf{C}\left(\mathbf{W}'_\mathbf{X}\mathbf{P}\right)^{-1}\mathbf{W}'_\mathbf{X}\right]\mathbf{x}_{\text{pred}}.$$

Dual

Let us define $\mathbf{A}_\mathbf{X}$ as the matrix containing $\alpha_{\mathbf{X},i}$ as its columns. Now we use the properties in Eqs. (26) and (23), showing that $\mathbf{W}_\mathbf{X} = \mathbf{X}'\mathbf{A}_\mathbf{X}$ and $\mathbf{X}^{k+1}\mathbf{W}_\mathbf{X} = \mathbf{0}$ leading to $\mathbf{W}'_\mathbf{X}\mathbf{P} \propto \mathbf{W}'_\mathbf{X}\mathbf{X}'\mathbf{A}_\mathbf{Y} = \mathbf{A}'_\mathbf{X}\mathbf{K}_\mathbf{X}\mathbf{A}_\mathbf{Y}$, where the proportionality is an equality up to a diagonal normalization matrix $\mathbf{A}'_\mathbf{Y}\mathbf{A}_\mathbf{Y}$ on the right-hand side. Furthermore, using Eq. (24), it is seen that $\mathbf{E}'_\mathbf{Y}\mathbf{A}_\mathbf{Y} = \mathbf{0}$ and thus (from Eq. (28)) that with the same diagonal normalization matrix as proportionality factor (which will thus be cancelled out), $\mathbf{C} \propto \mathbf{C}\mathbf{A}'_\mathbf{Y}\mathbf{A}_\mathbf{Y} = \mathbf{Y}'\mathbf{A}_\mathbf{Y}$. This leads to the complete dual form of regression-PLS:

$$\mathbf{y}_{\text{pred}} = \left[\mathbf{Y}'\mathbf{A}_\mathbf{Y}\left(\mathbf{A}'_\mathbf{X}\mathbf{K}_\mathbf{X}\mathbf{A}_\mathbf{Y}\right)^{-1}\mathbf{A}'_\mathbf{X}\mathbf{X}\right]\mathbf{x}_{\text{pred}}.$$

Note that the entire algorithm only requires the evaluation of kernel functions, since $\mathbf{X}\mathbf{x}_{\text{pred}}$ also consists of inner products only (or equivalently kernel evaluations $k(\cdot,\cdot)$). Using this fact, the solution can be cast in the standard form of kernel-based pattern recognition algorithms:

$$\mathbf{y}_{\text{pred}} = \sum_i \beta_i k(\mathbf{x}_i, \mathbf{x}_{\text{pred}}), \tag{29}$$

where β_i are the columns of $\beta = \mathbf{Y}'\mathbf{A}_\mathbf{Y}\left(\mathbf{A}'_\mathbf{X}\mathbf{K}_\mathbf{X}\mathbf{A}_\mathbf{Y}\right)^{-1}\mathbf{A}'_\mathbf{X}$.

5 Classification: Fisher Discriminant Analysis (FDA)

Definitions

We first define some symbols necessary to develop the theory. Since these quantities are defined in general for uncentered data, first this general definition is given. Afterwards, when appropriate the simplified formula will be provided for centered data. The latter formulas are the ones used in this section.

- Mean (n is the total number of samples $\mathbf{x}_i$)

$$\mathbf{m} = \frac{1}{n}\sum_i \mathbf{x}_i.$$

- Class mean ($\mathcal{S}_k$ is the set of samples belonging to cluster k, and $n_k = |\mathcal{S}_k|$, the number of samples in cluster k; thus $n = \sum_k n_k$)

$$\mathbf{m}_k = \frac{1}{n_k} \sum_{i:\mathbf{x}_i \in \mathcal{S}_k} \mathbf{x}_i.$$

- Total scatter matrix

$$\mathbf{S}_{\mathrm{T}} = \sum_k \sum_{\mathbf{x}_i \in \mathcal{S}_k} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})'.$$

- Within-class k scatter matrix

$$\mathbf{S}_k = \sum_{\mathbf{x}_i \in \mathcal{S}_k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)'.$$

- Within-class scatter matrix

$$\mathbf{S}_{\mathrm{W}} = \sum_k \mathbf{S}_k. \tag{30}$$

- Between-class scatter matrix

$$\mathbf{S}_{\mathrm{B}} = \sum_k n_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})'.$$

For centered data (as we will assume in the remainder of this section), we get:

$$\begin{aligned} \mathbf{m} &= \mathbf{0}, \\ \frac{1}{n} \sum_k n_k \mathbf{m}_i &= \mathbf{0}, \\ \mathbf{S}_{\mathrm{T}} &= \sum_k \sum_{\mathbf{x}_i \in \mathcal{S}_k} \mathbf{x}_i \mathbf{x}_i' = \mathbf{X}'\mathbf{X} = \mathbf{S}_{\mathbf{XX}}, \\ \mathbf{S}_{\mathrm{B}} &= \sum_k n_k \mathbf{m}_k \mathbf{m}_k'. \end{aligned}$$

Finally, the following properties hold:

- $\mathbf{S}_{\mathrm{T}} = \mathbf{S}_{\mathrm{B}} + \mathbf{S}_{\mathrm{W}}$.
- When the number of classes is 2, they can be indexed as $+$ and $-$, and:

$$\mathbf{S}_{\mathrm{B}} = \frac{n_+ n_-}{n} (\mathbf{m}_+ - \mathbf{m}_-)(\mathbf{m}_+ - \mathbf{m}_-)'. \tag{31}$$

5.1 Cost Function

Fisher discriminant analysis (FDA) [10] is designed for discrimination between two classes, indexed by + and −. It finds the direction $\mathbf{w}$ along which the between-class variance divided by within-class variance is maximized:

$$\mathbf{w} = \text{argmax}_{\mathbf{w}} \frac{\mathbf{w}'\mathbf{S}_\text{B}\mathbf{w}}{\mathbf{w}'\mathbf{S}_\text{W}\mathbf{w}}. \tag{32}$$

Note that when $\mathbf{w}$ is a solution, $c\mathbf{w}$ with c a real number is a solution too. In fact, we are not interested in the norm of $\mathbf{w}$, but only in the direction it is pointing at. Thus, equivalently, we could optimize the constrained optimization problem

$$\begin{aligned} \mathbf{w} &= \text{argmax}_{\mathbf{w}} \mathbf{w}'\mathbf{S}_\text{B}\mathbf{w} \\ &\text{s.t. } \mathbf{w}'\mathbf{S}_\text{W}\mathbf{w} = 1. \end{aligned} \tag{33}$$

5.2 Primal

This optimization problem can be solved by differentiating the Lagrangian $\mathcal{L}(\mathbf{w}, \mu) = \mathbf{w}'\mathbf{S}_\text{B}\mathbf{w} - \mu\mathbf{w}'\mathbf{S}_\text{W}\mathbf{w}$ with respect to $\mathbf{w}$ and equating to zero:

$$\begin{aligned} &\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, \mu) = \mathbf{0} \\ &\quad \Rightarrow \mathbf{S}_\text{B}\mathbf{w} = \mu\mathbf{S}_\text{W}\mathbf{w}. \end{aligned} \tag{34}$$

This is again a generalized eigenvalue problem, with both $\mathbf{S}_\text{B}$ and $\mathbf{S}_\text{W}$ symmetric and positive semidefinite. We are interested in the dominant eigenvector.

Another way to get the same result is by maximizing the correlation between the data projected on a weight vector $\mathbf{w}$ with the labels $\mathbf{y}$ (for each sample being 1 or −1, depending on the class the sample belongs to) of the corresponding data points. This is in fact CCA, applied on the data vectors on the one hand, and the labels on the other hand:

$$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{\mathbf{Xy}} \\ \mathbf{S}_{\mathbf{yX}} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{w}_{\mathbf{X}} \\ w_{\mathbf{y}} \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{S}_{\mathbf{XX}} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{\mathbf{yy}} \end{pmatrix} \begin{pmatrix} \mathbf{w}_{\mathbf{X}} \\ w_{\mathbf{y}} \end{pmatrix},$$

from which $\mathbf{w}_{\mathbf{X}}$ can be solved as

$$\mathbf{S}_{\mathbf{XX}}{}^{-1}\mathbf{S}_{\mathbf{Xy}}\mathbf{S}_{\mathbf{yy}}^{-1}\mathbf{S}_{\mathbf{yX}}\mathbf{w}_{\mathbf{X}} = \lambda^2\mathbf{w}_{\mathbf{X}}.$$

To see that $\mathbf{w}_{\mathbf{X}} = \mathbf{w}$, note that for centered data $\mathbf{X}$ (so $\mathbf{m}$ is made equal to $\mathbf{0}$ by centering), $\mathbf{S}_{\mathbf{XX}} = \mathbf{S}_\text{T} = \mathbf{S}_\text{B} + \mathbf{S}_\text{W}$, $\mathbf{S}_{\mathbf{yy}} = n$ is a scalar, and $\mathbf{S}_{\mathbf{Xy}} = \mathbf{X}'\mathbf{y} = n_+\mathbf{m}_+ - n_-\mathbf{m}_-$. One can then show that $\mathbf{S}_{\mathbf{Xy}}\mathbf{S}_{\mathbf{yy}}^{-1}\mathbf{S}_{\mathbf{yX}} = \frac{4n_+n_-}{n^2}\mathbf{S}_\text{B}$, and thus

$$\begin{aligned} &\frac{4n_+n_-}{n^2}\mathbf{S}_\text{B}\mathbf{w}_{\mathbf{X}} = \lambda^2(\mathbf{S}_\text{B} + \mathbf{S}_\text{W})\mathbf{w}_{\mathbf{X}} \\ &\quad \Rightarrow \mathbf{S}_\text{B}\mathbf{w}_{\mathbf{X}} = \frac{\lambda^2}{\frac{4n_+n_-}{n^2} - \lambda^2}\mathbf{S}_\text{W}\mathbf{w}_{\mathbf{X}}. \end{aligned}$$

This is exactly the Fisher discriminant generalized eigenvalue problem, with $\mu = \frac{\lambda^2}{\frac{4n_+n_-}{n^2} - \lambda^2}$ and $\mathbf{w} = \mathbf{w}_\mathbf{X}$.

5.3 Dual

Define $\mathbf{y}_+$ as $(\mathbf{y}_+)_i = \delta_{y_i,1}$ and $\mathbf{y}_-$ as $(\mathbf{y}_-)_i = \delta_{y_i,-1}$ (where we use the Dirac delta $\delta_{i,j}$, which is equal to 1 if $i = j$ and to 0 if $i \neq j$). The dual can again be derived by using $\mathbf{w} = \mathbf{X}'\boldsymbol{\alpha}$:

$$\begin{aligned}
&\mathbf{S}_\mathrm{B}\mathbf{w} = \mu \mathbf{S}_\mathrm{W}\mathbf{w} \\
&\Downarrow \qquad \text{Eqs. (30), (31)} \\
&\tfrac{n_+n_-}{n}\mathbf{X}(\mathbf{m}_+ - \mathbf{m}_-)(\mathbf{m}_+ - \mathbf{m}_-)'\mathbf{X}'\boldsymbol{\alpha} \\
&= \mu\mathbf{X}\textstyle\sum_{k=+,-}\sum_{\mathbf{x}_i\in\mathcal{S}_k}(\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)'\mathbf{X}'\boldsymbol{\alpha} \\
&\Downarrow \\
&\tfrac{n_+n_-}{n}\mathbf{K}_\mathbf{X}\left(\tfrac{\mathbf{y}_+}{n_+} - \tfrac{\mathbf{y}_-}{n_-}\right)\left(\tfrac{\mathbf{y}_+}{n_+} - \tfrac{\mathbf{y}_-}{n_-}\right)'\mathbf{K}_\mathbf{X}\boldsymbol{\alpha} \\
&= \mu\mathbf{K}_\mathbf{X}\left(\mathbf{I} - \tfrac{1}{n_+}\mathbf{y}_+\mathbf{y}_+' - \tfrac{1}{n_-}\mathbf{y}_-\mathbf{y}_-'\right)\mathbf{K}_\mathbf{X}\boldsymbol{\alpha} \\
&\Downarrow \\
&\mathbf{M}\boldsymbol{\alpha} = \mu\mathbf{N}\boldsymbol{\alpha},
\end{aligned}$$

where we substituted $\mathbf{M} = \frac{n_+n_-}{n}\mathbf{K}_\mathbf{X}\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)\mathbf{K}_\mathbf{X}'$, and $\mathbf{N} = \mathbf{K}_\mathbf{X}\left(\mathbf{I} - \frac{1}{n_+}\mathbf{y}_+\mathbf{y}_+' - \frac{1}{n_-}\mathbf{y}_-\mathbf{y}_-'\right)\mathbf{K}_\mathbf{X}$.

For centered data as is assumed here, the projection of a test point $\mathbf{x}_\mathrm{test}$ onto the FDA direction corresponding to $\boldsymbol{\alpha}$ can again be computed as

$$\sum_{i=1}^{n}\alpha_i k(\mathbf{x}_i, \mathbf{x}_\mathrm{test}).$$

5.4 Multiple Discriminant Analysis (MDA)

While Fisher discriminant analysis is originally designed for the two-class problem, optimization of the very same cost function (Eqs. (32) and (33)) leading to the same generalized eigenvalue problem in Eq. (34) can be used for solving the multiclass problem (e.g. [9]). In that case, a few generalized eigenvector may be necessary to do the classification (typically the number of clusters minus one).

The intuition behind this is to maximize the total between-class covariance for a certain amount of within-class covariance. This amounts to maximizing the signal-to-noise ratio present in the projections of the samples onto the discriminant directions. Here, the distance between the projected clusters is the signal one is interested in, and the variance in the projections of the clusters is the noise. Interestingly, it has been shown that PLS also maximizes the between-class covariance when computed on a class indicator matrix $\mathbf{Y}$, however, this is done without considering the within-class covariance [4, 20]. Deriving the dual version of MDA can be done in a similar way as for FDA.

6 Spectral Methods for Clustering

Clustering is a standard problem in pattern recognition: identify groups of samples that supposedly belong to the same class, without any information on the class labels (unsupervised). The problem is often solved with classical algorithms of which the K-means algorithm is the best known. Most of these algorithms are designed for data with Gaussian class distributions. In many cases, however, this is an oversimplification. Furthermore, many well-known algorithms are based on a nonconvex optimization problem.

Therefore in recent years a significant amount of research has been carried out in the field of *spectral clustering* (SC) [2, 5, 8, 17, 18, 22, 26, 32]. The clustering problem is relaxed or restated, leading to efficient algorithms with a simple eigenvalue problem at the core. Furthermore, in general no Gaussianity assumptions are made.

Spectral clustering algorithms generally consist of three components: the computation of a suitable *affinity matrix*, expressing the similarities between the samples; an *eigenvalue problem* based on this affinity matrix, returning (eigen)vectors that reflect the cluster structure in the data; and a final step performing the *actual clustering*, based on these eigenvectors. In the next three subsections we will briefly go into each of these aspects.

6.1 The Radial Basis Function as the Kernel

Whereas standard clustering methods assume Gaussian class distributions (or make similar assumptions on the distribution), spectral clustering methods intend not to do this. In order to achieve this goal, the use of the Euclidian inner product as a similarity measure between the samples is avoided. Instead, the kernel trick can be used to implicitly compute an inner product between feature maps of the samples. More specifically, in spectral clustering algorithms, most often the radial basis kernel function (*RBF kernel*) is used as similarity measure:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right).$$

Note that for $\|\mathbf{x}_i - \mathbf{x}_j\| \ll \sigma$, the RBF kernel is $k(\mathbf{x}_i, \mathbf{x}_j) \simeq 1 - \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}$. Thus, locally, the RBF kernel is related to the Euclidian metric. On the other hand, for two points at a farther Euclidian distance from each other (that is, $\|\mathbf{x}_i - \mathbf{x}_j\| \gg \sigma$), we have that $k(\mathbf{x}_i, \mathbf{x}_j) \simeq 0$. The result is that the algorithm will not see if a group of points with a 'diameter' considerably larger than σ is Gaussianly distributed or not. Only for samples that are relatively close to each other, it will give an indication of how close exactly they are. This is desirable: it allows us to cluster samples that are stretched out in a nonlinear shape.

Even though, in spectral clustering methods, very often an RBF kernel is used, it is important to know that the similarity measure does not have to be positive definite; however, for most spectral clustering variants (such as the ones described in Sects. 6.2.1 and 6.2.2), it has to be *nonnegative* (which is indeed true for the RBF kernel). Because of the absence of the positive definiteness requirement, the matrix containing the similarities between the samples is usually called the *affinity matrix* in this context, instead of the *kernel matrix.* Besides the RBF kernel matrix, other affinity matrices are used in literature, such as the k-nearest neighbor affinity matrix. However, for uniformity in this chapter, here we will continue to use the term kernel matrix instead of affinity matrix, and denote it by $\mathbf{K}$.

As opposed to the techniques discussed in the previous sections, *in spectral clustering, usually the kernel/affinity matrix is* not *centered.* In case it is centered, we will denote this explicitly, here, by using $\mathbf{K}_c$.

6.2 Which Eigenvectors?

We will only give a brief overview of the methods available in the literature. All of them compute the eigenvectors of a (generalized) eigenproblem involving $\mathbf{K}$. We will outline two methods that represent a relaxation of a discrete optimization problem on a graph, and another method based on the alignment between two matrices. Every method described is derived for the two-cluster case. However, they appear to be extendible towards multicluster problems, by taking more than one eigenvector (often $k-1$ when there are k clusters).

6.2.1 Normalized Cut Cost

Shi and Malik [26] start from graph theoretic concepts. They relax the problem of finding the minimal *normalized cut cost (NCut)* of the graph, where nodes of the graph correspond to samples and the (positive) kernel entries are the weights (*affinities*) of the edges in between the nodes. Intuitively, an NCut is the total affinity between the clusters, normalized by the total affinity of each cluster with the entire sample. Mathematically, this is

$$\mathrm{NCut}(\mathbf{K}, \mathbf{y}) = \frac{\sum_{i,j:y_i=-y_j=1} K_{ij}}{\sum_{i:y_i=1} \sum_j K_{ij}} + \frac{\sum_{i,j:y_i=-y_j=-1} K_{ij}}{\sum_{i:y_i=-1} \sum_j K_{ij}}.$$

Thus, one looks for a label assignment $y_i \in \{1, -1\}$ such that $\mathrm{NCut}(\mathbf{K}, \mathbf{y})$ is minimized.

This problem can be proven to be equivalent to minimizing $\frac{\widetilde{\mathbf{y}}'(\mathbf{D}-\mathbf{K})\widetilde{\mathbf{y}}}{\widetilde{\mathbf{y}}'\mathbf{D}\widetilde{\mathbf{y}}}$ subject to $\widetilde{y}_i \in \{1, -\widetilde{y}\}$, and $\widetilde{\mathbf{y}}'\mathbf{D}\mathbf{1} = 0$, for some $\widetilde{y}$ and for $\mathbf{D} = \mathrm{diag}(\mathbf{K}\mathbf{1})$. When the discrete vector $\widetilde{\mathbf{y}}$ is replaced by a continuous vector α_i, so the problem is relaxed, an *approximation* for the unrelaxed problem solution can be found by solving the generalized eigenvalue equation:

$$(\mathbf{D} - \mathbf{K})\alpha = \lambda \mathbf{D}\alpha \quad \text{s.t.} \quad \alpha'\mathbf{D}\mathbf{1} = 0,$$

where one is interested in the vector α corresponding to the smallest eigenvalue λ while satisfying the constraint. One can show, however, that the constraint is satisfied for all of the generalized eigenvectors except for the one with smallest eigenvalue $\lambda = 0$ with corresponding generalized eigenvector $\alpha = \mathbf{1}$. Thus, one searches for the eigenvector with the smallest nonzero eigenvalue.

6.2.2 Average Cut Cost

Another approach discussed in [26] is based on a relaxation of the minimum *average cut cost (ACut)* problem. The ACut cost is the sum of the (positive) kernel entries corresponding to pairs of points belonging to different classes, normalized by the number of samples in both classes:

$$\mathrm{ACut}(\mathbf{K}, \mathbf{y}) = \frac{\sum_{i,j:y_i=-y_j=1} K_{ij}}{\sum_{i:y_i=1} 1} + \frac{\sum_{i,j:y_i=-y_j=-1} K_{ij}}{\sum_{i:y_i=-1} 1},$$

where again $y_i \in \{1, -1\}$. This is similar to the NCut problem, and gives rise to a similar eigenvalue problem to be solved after relaxation:

$$(\mathbf{D} - \mathbf{K})\alpha = \lambda\alpha.$$

The eigenvector α corresponding to the smallest nonzero eigenvalue will reflect the cluster structure of the data.

6.2.3 Alignment-Based Approach

The alignment-based method (proposed in [8]) is a relaxation of the problem to find a label assignment that maximizes the alignment between the label matrix and the centered kernel matrix $\mathbf{K}_c$:

$$\max_{\mathbf{y}} \; \mathbf{y}'\mathbf{K}_c\mathbf{y} \quad \text{s.t.} \quad y_i \in \{1, -1\}.$$

Since this problem would be combinatoric again, it is relaxed by replacing the discrete vector $\mathbf{y}$ with a continuous vector α

$$\max_{\alpha} \; \alpha'\mathbf{K}_c\alpha \quad \text{s.t.} \quad \|\alpha\| = n$$

for n samples. This corresponds to solving the eigenvalue problem:

$$\mathbf{K}_c\alpha = \lambda\alpha.$$

Here the *dominant* eigenvector contains the relaxed labels as its entries.

6.3 What to do With the Eigenvectors?

We have now discussed how to compute eigenvectors that reflect the clustering in some way. There are different methods to extract the final clustering from these eigenvectors. In general, one constructs a matrix $\mathbf{A} = (\boldsymbol{\alpha}_1 \boldsymbol{\alpha}_2 \cdots \boldsymbol{\alpha}_k)$ containing the eigenvectors as its columns. Then some traditional distance-based clustering is performed on the rows of $\mathbf{A}$ in this k-dimensional space, sometimes after normalizing all rows of $\mathbf{A}$ to unit length. For further reading on different possible approaches we refer to the literature, see e.g. [18, 22, 36].

7 Summary

Table 1 contains the cost functions optimized for most of the algorithms described in this chapter. Tables 2 and 3 give the primal and the dual eigenproblems to be solved in order to optimize these cost functions. These tables contain columns $\mathbf{M}$, $\mathbf{N}$, and $\mathbf{v}$, each indicating which matrices and eigenvector to use in the generalized eigenproblem of the form $\mathbf{Mv} = \lambda \mathbf{Nv}$.

Given this, we still need to know how to project test data on the directions found by solving these generalized eigenproblems. This is summarized as:

- projection of a test sample onto weight vector in primal space $\mathbf{w}$: $\mathbf{w}'\mathbf{x}_{\text{test}}$.
- projection of a test sample onto weight vector in feature space corresponding to the dual vector $\boldsymbol{\alpha}$: $\sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_{\text{test}})$.

8 Conclusions

Among the algorithms discussed in this chapter, there are a number of classic methods from multivariate statistics, such as PCA and CCA; some methods that are virtually unknown in that field but are hugely popular in specific application domains, such as PLS; and finally some methods that are typically the product of the machine learning community, such as the clustering methods presented here, and all the extensions based on the use of kernels. Despite coming from so many different fields, the algorithms clearly display their common features, and we have emphasized them by casting them in a common notation and with a common language. From those comparisons, and from the comparison with the family of kernel methods based on quadratic programming, it is clear that this approach based on spectral methods can be considered another major branch of the KM family. The duality that emerges here from SVD approaches naturally matches the duality derived by the Kuhn–Tucker Lagrangian theory developed for those methods, and the statistical study demonstrates similar properties as shown in [27] and [28].

Some properties of this class of algorithms are already extremely appealing to machine learning practitioners, while others still need research attention.

Table 1. Cost functions optimized by the different methods

PCA	Maximize variance	$\frac{\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}}{\mathbf{w}'\mathbf{w}}$
		$\mathbf{w}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}$ s.t. $\Vert\mathbf{w}\Vert^2 = 1$
	Minimize residuals	$\Vert(\mathbf{I} - \mathbf{w}\mathbf{w}')\mathbf{X}\Vert_F^2$
CCA	Maximize correlation	$\frac{\mathbf{w}_{\mathbf{X}}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_{\mathbf{Y}}}{\sqrt{\mathbf{w}_{\mathbf{X}}'\mathbf{S}_{\mathbf{XX}}\mathbf{w}_{\mathbf{X}}}\sqrt{\mathbf{w}_{\mathbf{Y}}'\mathbf{S}_{\mathbf{YY}}\mathbf{w}_{\mathbf{Y}}}}$
	Maximize fit	$\mathbf{w}_{\mathbf{X}}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_{\mathbf{Y}}$ s.t. $\Vert\mathbf{X}\mathbf{w}_{\mathbf{X}}\Vert^2 = \Vert\mathbf{Y}\mathbf{w}_{\mathbf{Y}}\Vert^2 = 1$
	Minimize misfit	$\Vert\mathbf{w}_{\mathbf{X}}'\mathbf{X} - \mathbf{w}_{\mathbf{Y}}'\mathbf{Y}\Vert^2$ s.t. $\Vert\mathbf{X}\mathbf{w}_{\mathbf{X}}\Vert^2 = \Vert\mathbf{Y}\mathbf{w}_{\mathbf{Y}}\Vert^2 = 1$
PLS	Maximize covariance	$\frac{\mathbf{w}_{\mathbf{X}}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_{\mathbf{Y}}}{\sqrt{\mathbf{w}_{\mathbf{X}}'\mathbf{w}_{\mathbf{X}}}\sqrt{\mathbf{w}_{\mathbf{Y}}'\mathbf{w}_{\mathbf{Y}}}}$
	Maximize fit	$\mathbf{w}_{\mathbf{X}}'\mathbf{S}_{\mathbf{XY}}\mathbf{w}_{\mathbf{Y}}$ s.t. $\Vert\mathbf{w}_{\mathbf{X}}\Vert^2 = \Vert\mathbf{w}_{\mathbf{Y}}\Vert^2 = 1$
	Minimize misfit	$\Vert\mathbf{w}_{\mathbf{X}}'\mathbf{X} - \mathbf{w}_{\mathbf{Y}}'\mathbf{Y}\Vert^2$ s.t. $\Vert\mathbf{w}_{\mathbf{X}}\Vert^2 = \Vert\mathbf{w}_{\mathbf{Y}}\Vert^2 = 1$
FDA	Maximize between-class to within-class covariance	$\frac{\mathbf{w}'\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}'\mathbf{S}_{\mathrm{W}}\mathbf{w}}$
		$\mathbf{w}'\mathbf{S}_{\mathrm{B}}\mathbf{w}$ s.t. $\mathbf{w}'\mathbf{S}_{\mathrm{W}}\mathbf{w}$
SC1	Normalized cut cost	$\frac{\sum_{i,j:y_i=-y_j=1} K_{ij}}{\sum_{i:y_i=1}\sum_j K_{ij}} + \frac{\sum_{i,j:y_i=-y_j=-1} K_{ij}}{\sum_{i:y_i=-1}\sum_j K_{ij}}$
SC2	Average cut cost	$\frac{\sum_{i,j:y_i=-y_j=1} K_{ij}}{\sum_{i:y_i=1} 1} + \frac{\sum_{i,j:y_i=-y_j=-1} K_{ij}}{\sum_{i:y_i=-1} 1}$
SC3	Alignment	$\mathbf{K}_{\mathrm{c}}$

Table 2. Primal forms (not for spectral clustering algorithms)

	$\mathbf{M}$	$\mathbf{N}$	$\mathbf{v}$
PCA	$\mathbf{S}_{\mathbf{XX}}$	$\mathbf{I}$	$\mathbf{w}$
RCCA	$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{\mathbf{XY}} \\ \mathbf{S}_{\mathbf{YX}} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{S}_{\mathbf{XX}} + \gamma\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{\mathbf{YY}} + \gamma\mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \mathbf{w}_{\mathbf{X}} \\ \mathbf{w}_{\mathbf{Y}} \end{pmatrix}$
PLS	$\begin{pmatrix} \mathbf{0} & \mathbf{S}_{\mathbf{XY}} \\ \mathbf{S}_{\mathbf{YX}} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{I}\ \mathbf{0} \\ \mathbf{0}\ \mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \mathbf{w}_{\mathbf{X}} \\ \mathbf{w}_{\mathbf{Y}} \end{pmatrix}$
FDA	$\mathbf{S}_B$	$\mathbf{S}_W$	$\mathbf{w}$

PLS, for example, is designed precisely to operate with input data that are high-dimensional and present highly correlated features, exactly the situation created by the use of kernel functions. The match between the two concepts is perfect, and in a way PLS can be better suited to the use of kernels than maximal-margin methodologies. Furthermore it is easily extendible towards multivariate regression. On the other hand, one of the major properties of support vector machines is not naturally present in eigenalgorithms: sparseness. Deliberate design choices can be made in order to enforce it, but the

Table 3. Dual forms

	$\mathbf{M}$	$\mathbf{N}$	$\mathbf{v}$
PCA	$\mathbf{K}$	$\mathbf{I}$	α
RCCA	$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{K_X}^2 + \gamma\mathbf{K_X} & \mathbf{0} \\ \mathbf{0} & \mathbf{K_Y}^2 + \gamma\mathbf{K_Y} \end{pmatrix}$	$\begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix}$
PLS	$\begin{pmatrix} \mathbf{0} & \mathbf{K_X K_Y} \\ \mathbf{K_Y K_X} & \mathbf{0} \end{pmatrix}$	$\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$	$\begin{pmatrix} \alpha_\mathbf{X} \\ \alpha_\mathbf{Y} \end{pmatrix}$
FDA	$\frac{n_+ n_-}{n}\mathbf{K_X}\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)$ $\cdot\left(\frac{\mathbf{y}_+}{n_+} - \frac{\mathbf{y}_-}{n_-}\right)'\mathbf{K_X}$	$\mathbf{K_X}\left(\mathbf{I} - \frac{\mathbf{y}_+\mathbf{y}_+'}{n_+} - \frac{\mathbf{y}_-\mathbf{y}_-'}{n_-}\right)\mathbf{K_X}$	α
SC1	$\mathbf{D} - \mathbf{K}$	$\mathbf{D}$	α
SC2	$\mathbf{D} - \mathbf{K}$	$\mathbf{I}$	α
SC3	$\mathbf{K}_\mathrm{C}$	$\mathbf{I}$	α

optimal way to include sparseness in this class of methods still remains an open question. Another important point of research is the stability and statistical convergence of general eigenproblems for finite sample sizes. For work on the stability of the spectrum of Gram matrices, we refer to [24] and [25].

The synthesis offered by this unified view has immediate practical consequences, allowing for unified statistical analysis and for unified implementation strategies.

Acknowledgements

Tijl De Bie is a research assistant with the Fund for Scientific Research Flanders (F.W.O.–Vlaanderen). Furthermore, his research is supported by: the Research Council KUL: GOA-Mefisto-666, GOA-Ambiorics; the FWO: G.0240.99 (multilinear algebra), G.0407.02 (support vector machines); the Belgian Federal Government: Belgian Federal Science Policy Office, IUAP V-22 (Dynamical Systems and Control: Computation, Identification, Modelling, 2002-2006). Roman Rosipal's research was supported by funding from the NASA CICT/ITSR/NeMC and IS/HCC programs.

References

1. S. Akaho. A kernel method for canonical correlation analysis. In: *Proceedings of the International Meeting of the Psychometric Society (IMPS2001)*. Springer, Berlin Heidelberg New York, Osaka, July 2001

2. Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In: *Proceedings of The 42nd Annual Symposium on Foundations of Computer Science (FOCS2001)*, Las Vegas, October 2001
3. F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002
4. M. Barker and W.S. Rayens. Partial least squares for discrimination. *Journal of Chemometrics*, 17:166–173, 2003
5. Y. Bengio, P. Vincent, and J.F. Paiement. Learning eigenfunctions of similarity: Linking spectral clustering and kernel PCA. Technical Report 1232, Département d'informatique et recherche opérationnelle, Université de Montréal, 2003
6. M. Borga, T. Landelius, and H. Knutsson. A Unified Approach to PCA, PLS, MLR and CCA. Report LiTH-ISY-R-1992, ISY, SE-581 83 Linköping, Sweden, November 1997
7. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, Cambridge, 2000
8. N. Cristianini, J. Shawe-Taylor, and J. Kandola. Spectral kernel methods for clustering. In: T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems 14.* MIT Press, Cambridge, MA, 2002
9. R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification,* 2nd edn, Wiley, New York, 2001
10. R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, Part II:179–188, 1936
11. C. Fyfe and P. L. Lai. ICA using kernel canonical correlation analysis. In: *International workshop on Independent Component Analysis and Blind Signal Separation (ICA2000)*, Helsinki, June 2000
12. R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, Cambridge, 1985
13. R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, Cambridge, 1991
14. A. Höskuldsson. Pls regression methods. *Journal of Chemometrics*, 2:211–228, 1988
15. H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:321–377, 1936
16. I. T. Jolliffe. *Principal Component Analysis.* Springer, Berlin Heidelberg New York, 1986
17. R. Kannan, S. Vempala, and A. Vetta. On clusterings: good, bad and spectral. In: *Proc. of the 41st Foundations of Computer Science (FOCS2000)*, Redondo Beach, November 2000
18. A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In: T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems 14.* MIT Press, Cambridge, MA, 2002
19. R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of Machine Learning Research*, 2:97–123, 2001
20. R. Rosipal, L.J. Trejo, and B. Matthews. Kernel PLS-SVC for linear and nonlinear classification. In: *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pp. 640–647, Washington DC, 2003
21. B. Schölkopf and A. Smola. *Learning with Kernels.* MIT Press, Cambridge, MA, 2002

22. G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two patterns. In: *Proceedings of the Royal Society London B*, 224:21–26, 1991
23. J. Shawe-Taylor and N. Cristianini. *Kernel methods for Pattern Analysis*. Cambridge University Press, Cambridge, 2004
24. J. Shawe-Taylor, N. Cristianini, and J. Kandola. On the concentration of spectral properties. In T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, 2002
25. J. Shawe-Taylor, C. Williams, N. Cristianini, and J. S. Kandola. On the eigenspectrum of the gram matrix and its relationship to the operator eigenspectrum. In: *Proceedings of the 13th International Conference on Algorithmic Learning Theory (ALT2002)*, pp. 23–40, 2002
26. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000
27. J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002
28. J. A. K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor. A support vector machine formulation to PCA analysis and its kernel version. *IEEE Transactions on Neural Networks*, 14(2):447–450, 2003
29. V. N. Vapnik. *The Nature of Statistical Learning Theory*, 2nd edn., Springer, Berlin Heidelberg New York, 1999
30. J.-P. Vert and M. Kanehisa. Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA. In: S. Becker, S. Thrun, and K. Obermayer (eds.), *Advances in Neural Information Processing Systems 15*. MIT Press, Cambridge, MA, 2003
31. A. Vinokourov, N. Cristianini, and J. Shawe-Taylor. Inferring a semantic representation of text via cross-language correlation analysis. In: T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.) *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, 2002
32. Y. Weiss. Segmentation using eigenvectors: A unifying view. In: *Proceedings of the 7th International Conference on Computer Vision (ICCV1999)*, pp. 975–982, Kerkyra, September 1999
33. H. Wold. Path models with latent variables: The NIPALS approach. In: H.M. Blalock et al. (eds.), *Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building*. pp. 307–357. Academic, NY, 1975
34. H. Wold. Partial least squares. In S. Kotz and N.L. Johnson (eds.), *Encyclopedia of the Statistical Sciences*, vol. 6. pp. 581–591. Wiley, New York, 1985
35. Y. Yamanishi, J.-P. Vert, A. Nakaya, and M. Kanehisa. Extraction of correlated gene clusters from multiple genomic data by generalized kernel canonical correlation analysis. *Bioinformatics*, 19:323i–330i, 2003
36. H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. In T. G. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, 2002