Dipartimento di Fisica

Features Selection for Gene Expression Data Analysis

by

Sofia Mosci

DIFI, Università di Genova
v. Dodecaneso 33, 16146 Genova, Italy
http://www.fisica.unige.it/

Università degli Studi di Genova

Dipartimento di Fisica

Dottorato di Ricerca in Fisica

Ph.D. Thesis in Physics

Features Selection for Gene Expression Data Analysis

by

Sofia Mosci

February, 2009

Dottorato di Ricerca in Fisica
Dipartimento di Fisica
Università degli Studi di Genova

DIFI, Univ. di Genova
via Dodecaneso 33
I-16146 Genova, Italy
http://www.fisica.unige.it/

Ph.D. Thesis in Physics

Submitted by Sofia Mosci
Dipartimento di Fisica
Università degli Studi di Genova
mosci@disi.unige.it

Date of submission: February 2009

Title: Features selection for gene expression data analysis

Advisor: Alessandro Verri
Dipartimento di Informatica e Scienze dell'Informazione
Università degli Studi di Genova
verri@disi.unige.it

Supervisor: Pierantonio Zanghì
Dipartimento di Fisica
Università degli Studi di Genova
nino.zanghi@ge.infm.it

Ext. Reviewer: Massimiliano Pontil
Department of Computer Science
University College London
m.pontil@cs.ucl.ac.uk

Abstract

Gene expression analysis aims at identifying the genes able to accurately predict biological parameters like, for example, disease subtyping or progression. While accurate prediction can be achieved by means of many different techniques, gene identification, due to gene correlation and the limited number of available samples, is a much more elusive problem. Small changes in the expression values often produce different gene lists, and solutions which are both sparse and stable are difficult to obtain. Building on the elastic-net regularization strategy, we propose a new two-stage regularization method able to learn linear models characterized by a high prediction performance. By varying a suitable parameter these linear models allow to trade sparsity for the inclusion of correlated genes and to produce gene lists which are almost perfectly nested. Furthermore we develop a strategy that allows to obtain a more structured gene signature where genes are disposed in block of intra-correlated genes and the blocks are ranked according to a measure of their discriminative power. Experimental results on synthetic and microarray data confirm the interesting properties of the proposed method and its potential as a starting point for further biological investigations. Finally, motivated by the need of extending feature selection to nonlinear input/output dependencies, we develop a unified framework to characterize and solve, by means of an iterative projection algorithm, the optimization problems underlying a large class of sparsity based regression methods, encompassing lasso, group lasso and elastic net regularization, as well as more general sparsity based algorithms in reproducing kernel Hilbert spaces.

to my family

I don't believe in science, science is an intellectual dead end. Guys with tweed suits, cutting up frog on foundations grants,... (Woody Allen, *Sleeper*)

Table of Contents

Chapter 1

Introduction

This thesis presents a robust statistical analysis protocol able to select nested sets of relevant variables within the learning from examples framework. The proposed protocol is then applied to and further developed to fit the analysis of high-throughput gene expression data, where it allows to extract gene signatures, organize them in ordered modules of correlated genes ranked according to their prediction power, and efficiently visualize such a structure, hence easing the search of interesting biological patterns. The work can be appreciated for two main reasons. First, we provide theoretical motivations for performing the feature selection step, moreover, differently from heuristically motivated methods, the employed selection technique is well set in a robust theoretical framework. Second, also thanks to the particular attention given to intrepretability of the results, the proposed approach has large practical relevance in the treatement of cancer as well as other deseases, since an accurate molecular mechanisms understanding represents the first step toward effective targeted therapies.

Machine learning in gene expression data analysis

In order to better appreciate the results of this work, let us briefly introduce a few concepts concerning machine learning and its application to gene expression data analysis, a fast-growing research field, related to medicine and biology.

Learning from examples and feature selection

Machine learning is concerned with building automated systems that improve their performance on specific tasks by accumulating and processing experience. It is a subfield of Artificial Intelligence and intersects with statistics, cognitive science, information theory, and probability theory, among others. Supervised learning, or learning-from-examples, is a machine learning technique for finding a description of an unknown dependency between measurements of objects, inputs, and certain properties of these objects, called output. The purpose of estimating the dependency between the input and output variables is to be able to determine the values of output variables for any possibly unseen object. The problem of estimating an unknown dependency occurs in various practical applications. For example, the input variables can be the prices for a set of stocks and the output variable the direction of change in a certain stock price. As another example, the input can be some medical parameters and the output the probability of a patient having a certain disease. An essential feature of statistical learning is that the information is assumed to be contained in a limited set of examples (the sample), and the estimated dependency should be as accurate as possible for all objects of interest. In this thesis in particular we are interested in the subfield of learning from examples known as feature selection, which deals with the problem of extracting a subset of relevant variables from a supervised learning task.

Gene-expression microarrays

Gene-expression microarrays make it possible to simultaneously measure the rate at which a cell or tissue is expressing – translating into a protein – each of its thousands of genes. These comprehensive snapshots of biological activity can be used to infer regulatory pathways in cells, identify novel targets for drug design, and improve diagnosis, prognosis, and treatment planning.

Nevertheless, high-throughput technologies generate myriads of intricated data, and the analysis of these data presents unprecedented analytical and computational challenges. One the one hand, because of ethical, cost, and time constraints, most life science studies include a modest number of cases. On the other hand, modern high-throughput experiments measure several thousands varaibles per case. This is known as the curse of dimensionality. In genomic data sets, the number of variables can be in the order of 10^4, whereas the number of cases hardly exceeds the hundred, and is often in the order of few tens. These challenges have prompted scientists from a wide range of disciplines to work together towards the development of novel methods to analyze and interpret high-throughput data in genomics. In this context, the need for automated analysis of microarray data offers an opportunity for machine learning to have a significant impact on biology and medicine. Among the different tasks that can be faced in gene expression analysis a recent prominent application of machine learning to gene-array data is the feature selection task concerned with the identification of the genes able to accurately predict biological parameters like, for example, disease subtyping or progression. Moreover, in the context of microarray gene expresion analysis, an other interesting problem is devising tools to interpret the statistical results and understand their biological meaning.

Contributions

The main contributions of this thesis can be summarized as follows:

Theoretical motivations for performing dimensionality reduction We prove that, when empirical risk minimization is applied to the data projected onte the first m principal components, i.e. the directions of highest variance, indeed the number m *is* a regularization parameter and that an optimal parameter choice exists. Using probabilistic estimates for integral operators we can prove error estimates for this two-steps technique and propose a parameter choice procedure allowing to prove consistency of the algorithm. This result, originally presented in [83], can be appreciated for the theoretical motivations given to the otherwise heuristic process of dimensionality reduction.

Robust feature selection protocol for extracting nested lists of relevant variables Leveraging on the elastic-net regularization strategy [122], and exploiting recent theoretical results [30], we propose a two-stage method which produces variable subsets able to effectively address prediction problems from high-throughput data. In the first stage, the method learns from the available data a minimal set of variables which are best suited to accurately predict the parameter related to the problem at hand. By selecting the model through the combination of two optimization schemes, elastic net and regularized least squares, our method leads to a model which, unlike the elastic net alone, is characterized by both sparsity and low bias. In the second stage, by varying a suitable parameter, the method is able to produce models of increasing cardinality by gradually including variables correlated with the set of variables identified in the first stage. The two-stage procedure was initially presented in [31] and then rielaborated in [9] to deal with lowe size data sets.

Unifying picture of sparse regularization algorithms Under fairly mild assumption on the penalty (it must be one-homogeous), we develop a general framework to characterize and solve the optimization problems underlying a large class of sparsity based regularization algorithms, encompassing lasso, elastic net, group lasso and multiple kernel learning. Leveraging on convex analysis tools, we exploit the convexity of the objective functional to: (1) derive optimality conditions for the regularized solutions of several sparsity based methods; and (2) propose a general iterative projection algorithm for which we prove convergence to the regularized solution. Interestingly, the iterative soft thresholding method recently proposed to solve the Lasso minimization can be recovered as a corollary of our results, that are flexible enough to account for a large class of problems. Besides this first result, we use this fact to derive new extremely simple optimization schemes for multi-task learning, multi-kernel learning, sparse principal component analysis and total variation regularization among others. This work, still ongoing, was initially introduced in [85], whereas its application to the multi-kernel learning problem was presented in [84].

Experimental procedure for extracting modules of correlated relevant genes We show how the two-stage procedure can be efficiently applied to microarray data in order to select nested lists of relevant genes Furthermore we refine such a raw structure by means of a customized hierarchical agglomerative procedure based on Pearson correlation, hence identifying ordered modules of correlated genes.

Outline of the Thesis

The thesis is structured in two parts. In **Part I**, which deals with the problem of feature selection in the supervised learning setting, encompasses the following chapters:

Chapter 2 introduces the problem of dimensionality reduction with particular attention to the subproblem known as feature selection. After, examining its main motivations, we recast the problem of dimensionality reduction in the supervised learning framework, and identify two main subproblems, feature extraction and feature selection. We briefly review the process of feature extraction, and then concentrate on feature selection, providing a formal definition, and reviewing state-of-the-art tecniques organized in filters, wrappers and embedded methods.

In **Chapter 3** new theoretical results are presented that show that dimensionality reduction can be used as a tool to improve prediction accuracy, hence discrediting the preconception that the reduction of dimensionality is a necesessary evil, which allows for a reduced computational and storage cost, in spite of loss of relevant information. To do so, we provide theoretical and empirical evidence for performing dimensionality reduction based on its regularization properties when followed by a supervised learning step. In particular we prove consistency of the kernel principal component regression algorithm, where emprirical risk minimization is performed on the data projected, by means of Principal Component Analysis, on the m leading directions, the number of component m being the regularization parameter. As a main mathematical tool we use estimates of integral operators based on vector valued law of large numbers, to derive the probabilistic error estimates which are the keys to understand the role of m. Indeed such error estimates are made by two error terms, sample and approximation errors, and the best choice for m is the one balancing out the two terms. We then present some numerical results to illustrate and confirm the behavior of principal component regression on real and simulated data. We conclude the chapter discussing similar results, concerned with the specific problem of feature selection.

Chapter 4 deals with the feature selection problem. In particular, we propose an innovative two-stage regularization method, for extracting nested lists of relevant variables within a supervised learning framework. The selection core is a double optimization, based on ℓ^1(-ℓ^2) regularization followed by pure ℓ^2 regularziation. The proposed method is able to learn sparse linear models characterized by a high prediction performance, thanks to the double optimization, and extremely good stability of the selected features. Unlike other heuristically motivated methods, for this approach the consistency of the obtained estimator is guaranteed. Moreover another appealing property of the proposed approach is that the algorithm output is a one parameter family of nested lists with equivalent prediction ability and increasing correlation among variables.

In **Chapter 5** we develop a general framework to characterize and solve the optimization problems underlying a large class of sparsity based regularization algorithms, which amount to the minimization of a functional which is the sum of a convex differentiable term and a convex term which is not differentiable. Leveraging on convex analysis tools, in particular the theory of Fenchel duality and subdifferential calculus, we exploit the convexity of the functional to derive optimality conditions for the regularized solutions of several sparsity based methods; and especially to propose a general iterative projection algorithm for which convergence to the regularized solution is proved. The choice of a specific regularization will correspond to a particular projection operation and we discuss several examples encompassing, lasso, group lasso and elastic net regularization as well as more general sparsity based algorithms in reproducing kernel Hilbert spaces.

Part II deals with the application of the feature selection procedure developed in Part I to the analysis of microarray gene expression data, and is structured as follows:

Chapter 6 deals with the problem of selecting subsets of relevant genes from gene expression data, measured via microarrays. Following an introduction to microarray technology and stat-of-the-art on gene selection, we propose an experimental protocol, for extracting modules of correlated relevant genes from microarray data within a supervised learning framework. The gene selection core is the two-stage regularization method described in Chapter 4, which provides nested gene lists. Selection is then followed by a refinement of such a nested structure, leading to modules of correlated genes, that can be used by biologists as a tool in the interpretation process, possibly leading to new biological hypotheses that can represent the starting point for further biological investigations.

In **Chapter 7** we performed extensive experiments on real microarray data sets. We first analyze publicly available microarray datasets, concerning classification tasks. The data sets under consideration comprise one data set of in vitro samples, and three sets of data from in vivo tissue of three different tasks on three diseases. We then challenge our experimental protocol on a new medical trial, in collaboration with the children hospital Giannina Gaslini in Genova.

Part I

Machine Learning and Feature Selection

Chapter 2

Dimensionality Reduction

Dimensionality reduction is one of the most interesting problems in machine learning related to high-throughput data. In fact, advances in data collection and storage capabilities during the past decades have led to an information overload in most sciences. On the one hand, such data sets bring interesting opportunities, but, in contrast with smaller, more traditional data sets that have been studied extensively in the past, present new mathematical challenges in data analysis, and are bound to give rise to new theoretical developments. A common feature of high-dimensional data sets is that, in most cases, not all the stored information is "relevant" for understanding the underlying phenomena of interest. The aim of this chapter is thus to provide an introduction to the problem of dimensionality reduction with particular attention to the subproblem known as feature selection.

In Section 1 we introduce the problem of dimensionality reduction and investigate its main motivations. We then recast the problem in the supervised learning framework, and identify two main subproblems, which are known in the literature as feature extraction and feature selection.

In Section 2 we analyze the problem of feature extraction, i.e. the process of creating a more informative representation of the data. We provide a formal definition of the problem and review the most popular tools for performing feature extraction, grouped in intrinsically unsupervised and intrinsically supervised techniques.

In Section 3 we describe the problem of feature selection. After presenting a formal definition, we review state-of-the-art feature selection techniques, presented according to a well-established taxonomy which organizes these techniques in filters, wrappers and embedded methods.

2.1 The Problem of Dimensionality Reduction

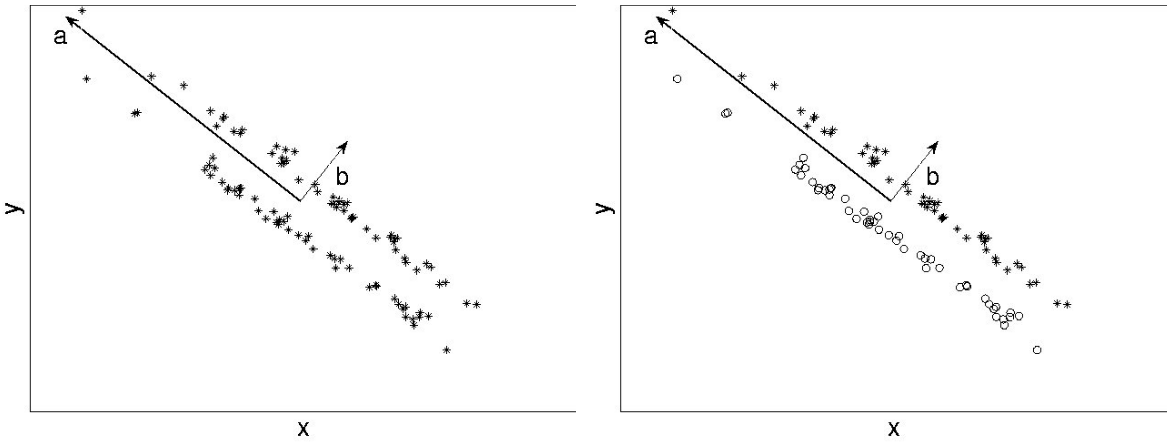
The *curse of the dimensionality* (term coined by Bellman in 1961 [15]) refers to the fact that, in the absence of simplifying assumptions, the sample size needed to estimate a function of several variables to a given degree of accuracy (i.e. to get a reasonably low-variance estimate) grows exponentially with the number of variables. A way to avoid the curse of the dimensionality is to reduce the input dimension of the function to be estimated; this is the basis for the use of dimensionality reduction both in unsupervised and supervised learning tasks.

More precisely, dimensionality reduction is the process of reducing the number of variables under consideration, given the fundamental assumption that the "relevant" data lies, at least approximately, on a manifold (nonlinear in general) of smaller dimension than the data space. The goal of dimensionality reduction is therefore to find a representation of such a manifold (a coordinate system) that will allow to project the data on it and to obtain a lower-dimensional, more compact representation that captures the content present in the original data. Such a definition is still incomplete in that it lacks a definition of the concept "relevance". Indeed, in the context of machine learning, we can identify two main interpretations of relevance according to the application domain. When only the input data are given we are in the unsupervised learning setting, where a main goal is therefore to reveal the structure of the input data, such as clusters or even more complicated patterns; in this context the relevant coordinates are identified with those coordinates which are most necessary for efficiently describing the underlying structure. On the other hand, the goal of supervised learning, where the inputs are presented with their corresponding outputs, or labels, is to infer the input-output relation in order to learn to produce the correct output given a new input; relevance in supervised learning applications is thus meant in terms of prediction ability.

To better clarify the difference of the two applications of dimensionality reduction, we can analyze two well-established techniques from the two different applications domains: Principal Component Analysis (PCA), probably the most common technique for dimensionality reduction in the unsupervised setting, and Linear Discriminant Analysis (LDA), a popular technique used in binary classification to enhance the class discriminatory information in a lower-dimensional space. Without going into details, we just need to know that while PCA seeks to optimally represent the data in terms of minimal reconstruction error, i.e. the mean-square-error between the representation and the original data, hence projecting onto the first eigenvectors of the covariance matrix of the inputs, LDA projects onto the directions that best separate the two classes, by maximizing the ratio of inter-class scatter over in-class scatter. For sake of clarity we can inspect the effect of the two techniques on the data set described in Figure 2.1. The inputs are described by two normally distributed variables, x and y. The eigenvectors of the covariance matrix are clearly identified with the two vectors $\mathbf{a}$ and $\mathbf{b}$. PCA will project onto $\mathbf{a}$ hence capturing the direction of highest variance. However, if the classes are distribuited as in Figure 2.1 right, the projection onto $\mathbf{a}$ causes the two classes to be indistinguishable. On the other hand, LDA will succeed in recognizing $\mathbf{b}$ as the most discriminative direction.

In general, there are several reasons for performing a reduction of dimensionality; in most cases such a process is carried out in order to achieve one or more of the following goals:

- **data compression** dictated by practical feasibility, such as limited storage capability, and processing time,



- improvement in **data visualization** for exploratory data analysis where a main goal is to reveal or enhance the class structure of the data through a visual inspection,
- improvement of **prediction accuracy** by discarding noisy components, hence limiting noise propagation,
- achievement of a **deeper understanding** of the underlying process, by revealing the input-output dependencies.

The first two items are typically related to unsupervised problems, since no knowledge on the outputs is involved. On the contrary the third and fourth goals characterize supervised learning tasks, where both input and output are given. In the rest of this thesis we restrict to the supervised learning setting, and therefore it is interesting to analyze more in details the last two motivations.

Concerning the third motivation in the next chapter we will provide theoretical and empirical evidence of the positive effect of dimensionality reduction on prediction accuracy. This concept is hardly understood if not even rejected since the reduction of dimensionality is often regarded as a necessary process, dictated by time and storage limitations, that causes a loss of potentially relevant information. The preconception that the more information we keep the more accurate the prediction we obtain is misleading, since, as we will show, keeping too much information does not necessarily improves prediction accuracy. Indeed some “shaving” of the data proves to have the same properties of other regularization techniques, hence avoiding unnecessary noise propagation and improving prediction performance.

Let us now analyze the last motivation. The use of dimensionality reduction as a regularization tool is motivated in those cases where the large size of the space of possible solutions requires some restriction in order to obtain statistical consistency of an estimator. However in many supervised learning tasks, the reduced size of the hypotheses space, that is the space of possible solutions, such as with linear kernel, already accounts for regularization. Therefore, though accurate prediction can be achieved without resorting to regularization, stability of the solution is not guaranteed. Moreover the identification of the relevant features involved in the underlying input-output relation represents the main target in many learning tasks. This is the case of biomedical data analysis, in particular of gene expression data from microarray, where biomarkers identification and a deeper understanding of the underlying biological process represent the major goals.

2.1.1 Dimensionality Reduction as a supervised learning problem

From now on we will analyze the problem of dimensionality reduction in the context of supervised learning. We therefore consider the setting of supervised learning, or learning-from-examples, where we have to find an unknown input-output relation given a finite number of input-output pairs. Such pairs are assumed to be identically and independently sampled according to an unknown probability measure $\rho(x, y) = \rho(y|x)\rho_X(x)$, where $x \in \mathcal{X} \subset \mathbb{R}^d$ and $y \in \mathcal{Y} \subset \mathbb{R}$. Differently, from classic statistics we are not interested in recovering the entire distribution ρ, but we aim at modeling the input-output relation in terms of a multivariate function $f : \mathcal{X} \to \mathcal{Y}$ such that $f(x) \sim y$, i.e. such that the error we commit when we predict the output or label y associated to a given input x with $f(x)$ is small in probability. Clearly the concept of small is qualitative and we therefore have to introduce a loss function $l : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}^+$ which quantifies the error. The supervised learning task can then be formalized saying that we aim at approximating the function f_ρ which minimizes the expected risk

$$\mathcal{E}(f) := \int_{\mathcal{X} \times \mathcal{Y}} l(y, f(x)) d\rho(x, y).$$

within a class of possible solutions $\mathcal{H}$, called the hypothesis space.

Note that the problem of inferring f_ρ is a much simpler task then the problem of recovering the entire probability distribution ρ. For this reason machine learning techniques succeed in many problems where most standard statistical tools fail. However in supervised learning tasks one has to deal with two complications: sampling and noise. One the one hand, the supervised learner has to correctly predict the value of the function f_ρ for any valid input after having seen a limited number of training examples. On the other hand, due to the probabilistic flavor of the learning setting, the measurements are noisy in the sense that the probability of measuring a pair (x, y) such that $y \neq f_\rho(x)$ is not zero. Indeed if $y = f_\rho(x)$ for all x we are in the pure deterministic setting where $\rho(x, y) = \delta(y, f_\rho(x))$.

In the context of supervised learning the two main directions for performing dimensionality reduction are known as **feature extraction** and **feature selection**. Feature extraction is the process of creating a representation for, or a transformation from the original data. In this case, even though such a representation can be lower dimensional, the extracted features can potentially depend on all original variables. On the other hand, in feature selection part of the input features are completely discarded, and thus only a (small) subset of the variables is kept. The next two sections are dedicated to a detailed analysis of such two learning processes.

2.2 Feature Extraction

Given a coordinate transformation $\psi(t,x) = (\psi_1(t,x), \ldots, \psi_p(t,x))$, where t denotes the hyper-parameter which defines the transformation, the problem of feature extraction, also called feature generation, or construction in the literature, amounts to the minimization of the risk

$$\mathcal{E}(t,f) := \int_{\mathcal{X}\times\mathcal{Y}} l(y, \psi(t,x)) d\rho(x,y).$$

with respect to f and the hyper-parameter t. The extracted features then identify a manifold which captures all the relevant information.

Even though applied in supervised learning, many methods for feature extraction are intrinsically unsupervised, in the sense that the features generation and ranking process involves only the input data, whereas the outputs are typically employed only in the selection of the number of features to be used for prediction. We hence review traditional state-of-the-art in feature extraction distinguishing between intrinsically supervised and unsupervised methods, hence avoiding the standard linear/nonlinear differentiation often followed in many books and surveys of the topic. In fact, most linear techniques for feature extraction can be easily extended to the nonlinear case by means of kernel methods, and we will then mention it when possible.

2.2.1 Intrinsically unsupervised feature extraction techniques

As mentioned above, in many feature extraction techniques, the features are extracted and ranked by observing only the input variables. The outputs are involved only subsequently in the choice of the optimal number of features to be kept for classification or regression. From the supervised learning perspective this approach is explained with the a priori assumption that the most discriminative features coincide with the "dominant" directions in the unsupervised sense.

PCA When the the relevance of a feature is measured via its variance, the features extraction problem has an exact analytical solution and corresponds to principal component analysis, PCA [59], probably the most widespread dimension reduction technique. PCA is based on the fact that the optimal approximation in the least square sense of a random vector $x \in \mathbb{R}^d$ by a linear combination of p independent vectors, with $p << d$, is obtained by projecting the random vector x onto the eigenvectors v_j corresponding to the largest eigenvalues λ_j of the covariance matrix of x, Σ_x. The first eigenvectors are then called the principal components. In practice the true covariance matrix Σ_x is not given, and PCA is performed on the empirical covariance matrix $X^T X$, where X is the $n \times d$ matrix which rows are the n realizations of the random vector x. If the assumption of unimodal gaussian distribution holds, PCA is able to find the independent axes of the data. In general, for non-Gaussian or multi-modal Gaussian data, PCA simply de-correlates the axes and hence builds the hyper-plane which minimizes the orthogonal distances to the data. As we will see in the next chapter, PCA can be used within the supervised learning framework, where it is named Principal Component Regression. In this context the labels y's can be used to decide how many principal components shall be kept to optimize prediction accuracy. PCA admits a straightforward extension to the non linear case, by substituting the covariance matrix with the kernel matrix.

ICA Independent Component Analysis, ICA (see [63] for details), has been introduced in order to solve the so called cocktail party problem, where the goal is to separate an observed multivariate mixture signals $x_1(t), \ldots, x_d(t)$ into additive subcomponents. These source signals (features) $s_1(t), \ldots, s_p(t)$ are assumed to be statistically independent, but, differently from PCA, not necessarily orthogonal to each other. Note that statistical independence of the features is a much stronger condition than uncorrelatedness, as required in PCA. While the latter only involves the second-order statistics, the former depends on all the higher-order statistics. In details, assume that we observe d linear mixtures $x_1, \ldots, x_d$ from d independent observers, $x_j(t) = a_{1j}s_1(t) + \ldots, a_{pj}s_p(t)$, which we can rewrite in matrix notation as $X = AS$. Our goal is to find a de-mixing matrix W such that $S = WX$. In order to solve this problem, a number of assumption must hold: first both mixture signals and source signals are zero-mean, then the sources have non-Gaussian distributions, finally the mixing matrix must be square, i.e., there are as many sources as mixing signals, $p = d$ (this assumption, however, can sometimes be relaxed). ICA is thus able to perform *blind source separation*, by exploiting independence and non-Gaussianity of the original sources.

Manifold Regularization Another important class of examples for intrinsically unsupervised features extraction is given by manifold regularization. This class of methods attempts to use the geometry of the marginal distribution, ρ_X, by assuming that its support has the geometric structure of a Riemannian manifold. Among manifold regularization techniques we recall **ISOMAP** [104], **Locally Linear Embedding** (LLE) [93], and Laplacian Regularization (LapRLS and LapSVM) [14]. In ISOMAP the aim is to find a transformation that preserves the geodesic distances between pairs of points in the high-dimensional space. Instead, LLE uses only distances within locally linear neighborhoods. Finally, in LapRLS and LapSVM a penalty term based on the graph Laplacian is added to the functionals which are minimized in Regularized Least Squares (RLS) and Support Vector Machine (SVM) respectively. The solution of the corresponding minimization problems should then reflects the intrinsic structure of the marginal probability ρ_X.

2.2.2 Intrinsically supervised feature extraction techniques

The main limitation of the class of techniques described above is that they do not explicitly consider class separability or prediction accuracy, since they do not take into account the labels of the feature vector. For instance, PCA simply performs a coordinate rotation that aligns the transformed axes with the directions of maximum variance. There is no guarantee that the directions of maximum variance will contain good features for discrimination. Therefore we now consider the other class of feature extraction techniques, where the outputs are explicitly taken into consideration during the extraction process.

Fisher Discriminant Analysis In the context of classification, among the intrinsically supervised feature extraction techniques, the most popular algorithm is probably Fisher Discriminant Analysis [45], which is available, as PCA, both in the linear version, where it is known as LDA and kernel version. In Fisher Discriminant Analysis the data are projected on the directions which provide the maximum separation between the classes means and the minimum variance within each projected class. In details Fisher Discriminant Analysis amount to maximizing the functional

$$J(w) = \{\frac{w^t S_B W}{w^T S_w w}\}$$

where S_W is the within-class scatter matrix, i.e. the sum of the two classes variance matrices, and $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)$ the between-class scatter (μ_1 and μ_2 are the means of the two classes in the original space). The maximizer of the optimization problem $w^* = \text{argmax}_w J(w)$ is known as Fisher's Linear Discriminant.

Partial Least Squares As Fisher Discriminant Analysis, also Partial Least Squares (PLS) [101], uses both inputs and outputs in order to construct a new set of features for regression, where each features is a linear combination of the input variables. In fact, in contrast to PCA, where the k-th principal component v_k solves

$$\max_{\|\alpha\|=1, v_l^T X^t X\alpha=0, \forall l \neq m} Var(X\alpha),$$

the k-th PLS direction ϕ_k solves:

$$\max_{\|\alpha\|=1, v_l^T X^T X\alpha=0, \forall l \neq m} Corr^2(Y, X\alpha)Var(X\alpha),$$

where X is the $n \times d$ inputs matrix and Y is the output vector, n being the number of samples and d the number of input variables. In short, in the construction of a set of linear combinations of the input variables, principal components analysis finds the variables weights reflecting the covariance structure between the new feature and the original variables, while partial least squares regression extracts the weights which best reflect the covariance structure between the new feature and the outputs.

2.3 Feature Selection

At the beginning of the previous section we have seen that the feature extraction problem amounts to the minimization of the risk with respect to a hyper-parameter t which encodes the coordinate transformation mapping the input variables into the new features. When such a transformation is differentiable with respect to t, and if the transform is smooth, then it is possible, at least in principle, to learn the transform by standard convex optimization techniques. In contrast, feature selection is not a smooth process, since switching from a feature subset to another is a discrete event, and thus requires different approaches. However, feature selection is preferable to transforms in those cases wherein it is essential to retain some of the original features. In addition, when the number of irrelevant features exceeds the number of relevant ones by orders of magnitude, learning a transform reliably may require excessive amounts of training data, whereas variable selection is a much more feasible task, even with low size data sets.

In this section we first provide a rigorous formal definition of the problem of feature selection. We then review the most important approaches and algorithms to solve such a problem.

2.3.1 Formal definition of Feature Selection

In many supervised learning problems the function f_ρ minimizing the expected risk may not depend on part of the input variables $x_1, \ldots, x_d$. The problem of feature selection is thus involved when we are interested in identifying which subset of features f_ρ depends on.

Let us introduce the set $\mathscr{I}$ containing all the possible subsets of $\{1, \ldots, d\}$. Given an element $\mathcal{I} \in \mathscr{I}$ we define the mapping $x \mapsto x_\mathcal{I}$ such that $[x_\mathcal{I}]_j = x_j 1(j)$ for all $j = 1, \ldots, d$, that is the operation that sets to zero all the components of x, x_j such that $j \notin \mathcal{I}$. The problem of selecting the most relevant features can then be formalized as the minimization of the expected risk

$$\mathcal{E}(\mathcal{I}, f) = \int_{\mathcal{X}\times\mathcal{Y}} l(y, (f(x_\mathcal{I}))d\rho(x, y). \tag{2.1}$$

Moreover, among all the pairs of minimizers $\{(\mathcal{I}^*, f^*)\} \subset \mathscr{I} \times \mathcal{H}$ of (2.1) we select the pair involving the minimum number of features, i.e.

$$(\mathcal{I}^\dagger, f^\dagger) = \underset{\{(\mathcal{I}^*, f^*)\}}{\operatorname{argmin}} |\mathcal{I}| \tag{2.2}$$

Such a double minimization does not guarantees uniqueness of the solution as a function on $\mathcal{H}$, because in general the minimization problem

$$\underset{f\in\mathcal{H}}{\operatorname{argmin}} = \int_{\mathcal{X}\times\mathcal{Y}} l(y, f(x))d\rho(x, y) = \int_{\mathcal{X}\times\mathcal{Y}} l(y, f(x))d\rho(y|x)\rho_X(x)$$

is not strictly convex when the support of the marginal distribution ρ_X is strictly contained in $\mathcal{X}$. In this case the homomorphism $h : \mathcal{H} \to L^2(X, \rho_X)$ is not bijective since we can find two different functions in $\mathcal{H}$ having the same image in $L^2(X, \rho_X)$.

In order to clarify this concept we provide a simple example. Let us consider a probability distribution on $\mathcal{X} = \mathbb{R}^2$ such that $\rho_X(x_1, x_2) = \delta(x_1, x_2)$ and the hypothesis space of the linear functions in $\mathbb{R}^2$. Clearly the support of ρ_X (i.e. set closure of the set of pairs (x, y) for which $f(x, y)$ is not zero) coincides with the bisector $x_2 = x_1$, so that if $f^*(x_1, x_2) = \beta_1^* x_1 + \beta_2^* x_2$ minimizes the risk, any other function $f(x_1, x_2) = \beta_1 x_1 + \beta_2 x_2$ such that $\beta_1 + \beta_2 = \beta_1^* + \beta_2^*$ is also a minimizer, because $f(x_1, x_2)\delta(x_1, x_2) = f^*(x_1, x_2)$. Notice that, due to the non invertibility of the homomorphism h, an optimal subset of features is not necessarily unique. For instance if we consider the above example an optimal feature subset can be $\{x_1\}$, $\{x_2\}$ or also $\{x_1, x_2\}$. Condition (2.2) restricts the choice to either $\{x_1\}$ or $\{x_2\}$, however there is no way to chose one of the two subsets of minimal cardinality, being both equivalent in terms of risk. In the case of biomedical data, in particular of microarray data, the problem of obtaining a unique solution to the feature selection task is not as crucial, as it will be discussed in Section 4.

Clearly, even though the problem of feature selection is easy to be understood in an abstract way, we have seen woe its mathematical formulation is far from being trivial. For this reason a lot of effort has been dedicated by the machine learning community to finding efficient solutions to this problem, as we will see in the rest of the chapter.

2.3.2 State-of-the-art in Feature Selection

A direct way of performing feature selection would be to extensively explore the features domain by trying out all the possible combinations of variables. However such a combinatorial problem is known to be NP hard [2], and thus completely hopeless as soon as we increase the number of variables. As a consequence, over the last decade many learning techniques have been proposed to approximate the problem of feature selection. We will follow the taxonomy proposed in [20], where feature selection techniques are grouped in three main classes: *filters* that "use feature selection to filter features passed to induction", *wrappers* that "treat feature selection as a wrapper around the induction process", *embedded* methods that "embed the selection within the basic induction algorithm".

2.3.2.1 Filters

The term *filter*, introduced by John, Kohavi, and Pfleger [66], denotes those feature selection techniques that filter out irrelevant attributes before learning is performed hence ignoring the effect of the filtered feature subset on the performance of the induction algorithm. Though in principle can be performed by means of many different techniques, a filter is typically identified with a fast preprocessing step, and is thus at risk of selection bias if filtering is performed before the data is split in training and test set.

Most filter approaches are based on ranking criteria, were the features are ordered and then selected or discarded according to a fixed threshold. The limit of filters method is that there is no guarantee that the ranking criterion used in the selection phase is optimal with respect to prediction which is the final goal of the supervised problem. Despite these drawbacks filters are still highly employed due to their simplicity that allows to remove a large number of features with extremely fast computations.

We now report some details about the most popular ranking methods, which differ mostly for the way they evaluate the features distinctiveness. For more details on the topic we refer to [115, 46, 114] and references therein.

Statistical scores Statistical tests evaluate the relevance of features based on the estimation of statistical scores and are very popular in classification problems especially in the analysis of biomedical data. As most ranking criteria they are simple and often effective, however they can fail in identifying the significant features when the assumption about the nature of the distributions does not hold. Many statistical tests are based upon the assumption that the data are sampled from a Gaussian distribution. These tests are referred to as parametric tests. Commonly used parametric tests are the t-test, the analysis of variance (ANOVA) and Pearson correlation. Tests that do not make assumptions about the population distribution are referred to as nonparametric tests or distribution-free tests. Wilcoxon test and all commonly used nonparametric tests rank the outcome variable from low to high and then analyze their ranks. We refer to [113] for a survey of the topic.

Entropy-based methods (Mutual Information) Another class of ranking criteria for feature selection are based on the information theoretic approach wherein the deviation from pure randomness is estimated by entropy of a distribution. In these methods a score is assigned to each feature based on its information gain. In order to quantify such a gain, most of these methods rely on empirical estimates of the mutual information between each variable and the output. For all j this is defined as:

$$I(j) = \int_{x_j} \int_{y} log \frac{p(x_j, y)}{p(x_j)p(y)} p(x_j, y) dx_j dy$$

where $p(x_j)$ and $p(y)$ are the probability densities of the j-th input variable x_j and the output y, and $p(x_j, y)$ is the joint density. The criterion $I(j)$ is a measure of dependency between the density of variable x_j and the density of the output y.

Single variable classifiers Another possible way to filter features is to use the performance of univariate classifier as a ranking criterion. The performance of the variable can be measured not only in terms of the error rate, but also by means of false positive classification rate or false negative classification rate. This approach leads to the selection of the variables having highest individual predictive power. However it is not guaranteed that the subset of most univariately predictive variables coincides with the most predictive variables subset.

2.3.2.2 Wrappers

Wrappers were introduced in opposition to filters in [66] where the authors pointed out the fact that the process of selecting features should depend not only on inputs and outputs, but also on the induction algorithm. Indeed, while in filtering features are selected independently of the learning machine, in wrappers the relevance of a feature subset is determined according to prediction performance of the learning algorithm itself. In fact, we can think of a filter where variables are ranked according to the prediction error of a single variable classifiers; indeed, if the final induction algorithm is multivariate, it is not guaranteed that the selected features are optimal with respect to the learning machine, because the most relevant features do not necessarily coincides with the most relevant feature subset. Wrappers were thus introduced to overcome such a bias.

In the wrapper approach the learning machine is used as a black box or a subroutine which, given in input a feature subset, returns a score according to its prediction performance. Therefore no knowledge of the machine is required, but only the ability to test its performance on a validation set is needed. The search algorithm is thus *wrapped* around the learning machine in order to explore the space of feature subsets. Wrappers were proposed as a computationally efficient heuristic alternative to the exhaustive search over the space of all possible feature subsets (2^d in the presence of d features). First, one must determine the starting point (or points) in the space, which in turn influences the direction of the search and the conditions used to generate successor states. This suggests that one might start with nothing and successively add attributes, or one might start with all attributes and successively remove them, as it is done in *forward selection* and *backward elimination* respectively. The former search engine starts from an empty set, and greedily incorporate the most relevant features into larger and larger subsets; the opposite procedure is performed through *backward elimination*, also known as *sequential backward selection*, where the least promising features are greedily eliminated, starting with the set of all the features. It is worth remarking that, since in feature selection the number of selected features is typically much smaller than the number of input features, forward selection could be preferable, in that it starts with small feature subsets, and is thus computationally cheaper if stopped early.

Despite the great improvement of greedy search with respect to exhaustive search, wrappers have still a much higher computational cost than filters, which results from calling the induction algorithm for each feature set considered. In fact, evaluating a new feature set in a wrapper method is done by internal validation methods, such as k-fold cross-validation or leave-one-out validation. For this reason ingenious techniques have been proposed for speeding up the evaluation process, see [20] and [69] for details.

2.3.2.3 Embedded methods

Differently from wrappers and filters, where variable selection and training are two separate processes, *embedded* methods present the advantage of incorporating feature selection within the construction of the classifier or regression model, i.e. as part of the training phase. Examples are decision trees [21], where a built-in mechanism is used to perform variable selection. In the last few years Adaboost [49, 59] and other variation of boosting [94, 51, 74] have also been proposed as embedded methods and empirical evidence of their effectiveness in several domains [111, 74, 7] has been reported. A sparsity enforcing penalty is also typical of many embedded methods minimizing an objective function defined as the penalized empirical error with a parameter balancing the sparsity of the solution. In the following we review the most important examples of embedded methods.

Decision trees Tree-based methods perform recursive binary partitions of the feature space, which is so clustered into a set of rectangles, and then fit a simple model (usually a constant) on each region. At each step the space is partitioned into two regions based on the value of a single variable. In regression, the variable and the split-point, or node, are chosen according to best fit - square-error node impurity measure, in classification the choice is dictated by misclassification error, or other classification measures, such as Gini index or cross-entropy. Decision trees can be efficiently used in feature selection, since only part of the input variables are used to partition the feature space. Variables that are not involved in any split are then discarded. In order to improve the classification rate, *random forest* were introduced where a number of decision trees are used.

Adaboost Boosting refers to a general and provably effective method of producing a very accurate prediction rule by combining rough and moderately inaccurate rules. Among boosting techniques AdaBoost is definitely the most popular *meta-algorithm*, which is used to boost the classification performance of a simple (sometimes called *weak* or *base*) learning algorithm.
To better understand the properties of AdaBoost, let us consider a binary classification task, with output variables coded as $y \in \{-1, 1\}$. Given a training set $(x_i, y_i), i = 1, \ldots, n$, the idea behind Adaboost is to apply the weak classifier many times on differently weighted versions of the training data, thus obtaining a sequence of weak classifiers. The predictions from all of them are then combined through a weighted majority vote to produce the final classification, where the effect of the weight is to higher influence to the more accurate base classifiers. The

weights given to each training sample are initialized to $1/n$, so that at the beginning each sample receives the same weight. The data modifications at each boosting step m consist in updating each sample weight, by giving an higher weight to examples misclassified at step $m-1$, and a lower weight to the others. This re-weighting is done according to the error rate of classifier C_{m-1}. In this way each successive weak classifier is forced to concentrate on those training data that are misclassified by the previous ones in the sequence.

Regularization with sparsity constraint Regularization is one possible key to perform embedded feature selection in the supervised learning framework, thanks to penalties which allow to enforce sparsity of the model, namely to perform automatic feature selection by assigning truly zero weights to all but a small number of selected features. In this kind of feature selection techniques a linear model is usually assumed $f_\beta(x) = \beta \cdot x$, and the solution is given by the minimizer of the functional:

$$\mathcal{E}_{\mathbf{z}}(f_\beta) + \lambda \Omega(\beta) \tag{2.3}$$

where the $\mathcal{E}_{\mathbf{z}}(f_\beta)$ is the empirical counterpart of the expected risk, $\Omega(\beta)$ is a penalizing term enforcing sparsity of the coefficient vector β, and $\lambda > 0$ is a trade-off coefficient balancing the empirical risk with this penalizing term. The most used penalty term is the ℓ^1-norm of the coefficient vector β, $\|\beta\|_1 = \sum_{j=1}^{d} |\beta_j|$. In order to find the minimizer of (2.3) with the least square error, Algorithms like LASSO an acronym for “Least Absolute Shrinkage and Selection Operator” [106], LARS [38] and basis pursuit [27] have been proposed in different contexts. Since feature selection by means of sparse regularization is the main topic this Ph.D. thesis, a more detailed review of the subject will be presented in Chapter 4.

Chapter 3

“The more information the better”: a wrong preconception

In many applications dimensionality reduction in general, and feature selection in particular, are often regarded as a necesessary evil, which allows for reduced computational and storage costs, in spite of loss of relevant information. This chapter is dedicated to showing that such preconception is incorrect. To this aim we will show that dimensionality reduction can be used as a tool to improve prediction accuracy. In fact, if the true statistical model is known, or if the training set is unlimitetd, any additional information can obviously only improve accuracy. However, when the sample is finite, additional features can degrade the prediction performance, even when all the features are statistically independent and carry information on the output. In this chapter we describe the work proposed in [83] providing theoretical motivations for performing dimensionality reduction based on its regularization properties when followed by a supervised learning step. To do so we investigate the effect of using Principal Compent Analysis as a preprocessing step in the prediction performance of empirical risk minimization performed on the projected data. We prove that, when empirical risk minimization is applied to the data projected onto the first m principal components, indeed the number m *is* a regularization parameter, and an optimal parameter choice exists, ensuring concistency of the estimator.

In Section 1 we introduce the setting, which is the usual statistical learning setting, where the design is random. We then give a brief overview of the theory of reproducing kernel Hilbert spaces, recalling the derivation of the empirical risk minimization solution on this classes of functions.

In Section 2 we provide theoretical and empirical evidence of the regularization properties of dimensionality reduction. In particular we first show that performing kernel PCA, and then ordinary least squares on the projected data is mathematically equivalent to truncated singular value decomposition or spectral cut-off regularization. In this contex the regularization parameter is identified with the number, m, of principal components to be used for prediction. Second, using probabilistic estimates for integral operators based on vector valued law of large numbers, we can evaluate error estimates and propose a parameter choice procedure allowing to prove consistency of the algorithm. Finally, we report numerical experiments that confirm theoretical results. We anticipate that in this section we will consider data-driven model selection via cross validation, rather than the optimal parameter choice provided by the theory, since in typical applications the bound we obtain, being basically distribution independent, will be too pessimistic.

We conclude Section 4, by reporting similar results concerning regularization properties of feature selection. Starting from the peaking phenomenon in controlled classfication tasks, we then recall the most important consitency results of feature selection via ℓ^1 regularization in the general statistical learning setting.

3.1 Machine Learning

Machine learning is a branch of statistics and computer science, which studies algorithms and architectures that learn from observed facts (examples). The main characteristic of machine learning systems is that they are trained instead of programmed, i.e. their goal is to extract some knowledge from training objects in order to generalize it to unseen situations. The kind of knowledge to be extracted can range from some information on the data distrubtion, such as clusters or other patterns, to certain properties of the objects, called output. The former situation is referred to as unsupervised learning, and the latter as supervised learning.

3.1.1 Supervised Learning

Supervised learning or learning-from-examples is a machine learning technique for finding an unknown input-output relation given a finite number of input-output instances. The output of the function can be a continuous value (*regression*), or can predict a class label of the input object (*classification*). For example, the input variables can be the prices for a set of stocks and the output variable the direction of change in a certain stock price. As another example, the input can be some medical parameters and the output the probability of a patient having a certain disease. Learning techniques are similar to fitting a multivariate function to a certain number of measurements data. However the main difference is that the task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples.

More precisely we assume that a training set $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = (x_1, y_1), \ldots, (x_n, y_n)$ is sampled according to an unknown distribution $\rho(x, y) = \rho(y|x)\rho_X(x)$, where $x \in \mathcal{X} \subset \mathbb{R}^d$ and $y \in \mathcal{Y} = [-M, M] \subset \mathbb{R}$. Notice that for the sake of simplicity we consider bounded outputs, though more general kind of noise - such as sub-Gaussian noise - can also be treated. The idea is to find a function f such that $f(x) \sim y$ and, according to the loss function $l : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$, this can be formalized saying that we look for a function with small expected error

$$\mathcal{E}(f) := \int_{\mathcal{X} \times \mathcal{Y}} l(y, f(x)) d\rho(x, y).$$

A natural choice for the loss function is the squared loss function $l(y, f(x)) = (y - f(x))^2$. In fact, one can easily show that, among all measurable functions the one that minimizes the expected error is the regression function

$$f_\rho := \int_{\mathcal{Y}} y d\rho(x, y).$$

Then, given a training set $\mathbf{z}$, the goal is to build an estimator $f_\mathbf{z}$ whose error is close to $\mathcal{E}(f_\rho)$. In particular a first important property is (weak) consistency

$$\lim_{n\to\infty} \Pr\left(\mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_\rho) \geq \epsilon \right) = 0 \qquad \forall \epsilon > 0$$

ensuring that, if we have enough data, we can eventually reach the best possible solutions for any probability distribution. A second crucial property concerns rate for the above convergence and is typically studied via probabilistic error estimates such that with probability at least $1 - \eta$

$$\mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_\rho) \leq \varepsilon(n, \eta) \tag{3.1}$$

where $\varepsilon(n, \eta)$ is a suitable bound depending on the number of samples and the confidence.

We also recall that for classification problems, i.e. $y \in \{-1, +1\}$, rather than the expected error we want to estimate the misclassification error

$$R(f) := Pr(yf(x) < 0)$$

whose minimizer R^*, namely the Bayes risk, is achieved by the Bayes rule

$$b(x) = \begin{cases} +1 & \text{if} \quad p(1|x) > \frac{1}{2} \\ -1 & \text{if} \quad p(1|x) \leq \frac{1}{2}. \end{cases} \tag{3.2}$$

In this case we wish to find a classification rule such that with probability at least $1 - \eta$ we have

$$R(f_\mathbf{z}) - R^* \leq \varepsilon(n, \eta)$$

and derive convergence of the misclassification error of our classification rule to the Bayes risk, namely Bayes consistency. Finally, we note that considering least squares estimates, a plug-in classification rule can be obtained taking $sign(f_\mathbf{z})$; moreover, since y is ±1, we get $f_\rho(x) = 2\rho(1|x) - 1$ so that the bayes rule is simply $sign(f_\rho)$. Interestingly the error measured via expected error and the misclassification error are related [10]

$$R(f) - R^* \leq \sqrt{\mathcal{E}(f) - \mathcal{E}(f_\rho)} \tag{3.3}$$

so that consistency w.r.t. to expected errors implies Bayes consistency.

3.1.2 Learning with Kernels

The search for possible solutions is often restricted to a hypotheses space $\mathcal{H}$. In the following we consider hypotheses spaces that are reproducing kernel Hilbert (RKH) spaces [6]. Recall that these are Hilbert spaces of functions which are completely determined by a symmetric positive definite function $K(x, s)$. In particular we make use of the following well-known properties:

- reproducing property: for $f \in \mathcal{H}$ it holds

$$f(x) = \langle f, K(x, \cdot) \rangle_{\mathcal{H}}\,; \tag{3.4}$$

- feature map: we can consider a mapping $\Phi : X \to \mathcal{H}$ which can be seen as a data parameterization related to the kernel through the following equality

$$\langle \Phi(x), \Phi(s) \rangle_{\mathcal{H}} = K(x, s), \quad x, s \in X.$$

For technical reasons we will assume the kernel to be continuous and bounded, i.e.

$$\kappa^2 = \sup_{x \in X} K(x, x) < \infty.$$

It is interesting to recall the derivation of the solution to empirical risk minimization (ERM) algorithm

$$f_{\mathbf{z}} = \operatorname*{argmin}_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2, \tag{3.5}$$

when $\mathcal{H}$ is a RKH space. If we consider the feature map

$$\Phi(x) = K(x, \cdot) =: K_x$$

the function in $\mathcal{H}$ can be written as $f(x) = \langle w, \Phi(x) \rangle$ and we can simply differentiate the empirical risk with respect to w to get a normal equation

$$\frac{1}{n} \sum_{i=1}^{n} \langle w, \Phi(x_i) \rangle_{\mathcal{H}} \Phi(x_i) = \frac{1}{n} \sum_{i=1}^{n} y_i \Phi(x_i).$$

Interestingly if the data are centered then we have that

$$T_{\mathbf{x}} := \frac{1}{n} \sum_{i=1}^{n} \Phi(x_i) \otimes \Phi(x_i) = \frac{1}{n} \sum_{i=1}^{n} \langle \cdot, \Phi(x_i) \rangle_{\mathcal{H}} \Phi(x_i) \tag{3.6}$$

is simply the (uncentered) covariance operator and the solution can be written as

$$w = T_{\mathbf{x}}^{\dagger} h_{\mathbf{z}} \tag{3.7}$$

with $h_{\mathbf{z}} = \frac{1}{n} \sum_{i=1}^{n} y_i \Phi(x_i)$ and $T_{\mathbf{x}}^{\dagger}$ denotes the generalized inverse of the covariance operator. We use this equation extensively in the next section, but we note here that from a practical point of view when the Hilbert space is *not* finite dimensional one usually prefers to use the fact that the solution can also be written as

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i)$$

where $\alpha = \mathbf{K}^{\dagger}\mathbf{y}$ and $\mathbf{K}^{\dagger}$ is the generalized inverse of the kernel matrix, $[\mathbf{K}]_{ij} = K(x_i, x_j)$.

3.2 Regularization properties of Principal Component Regression

Principal component analysis [59] is a very common statistical tool for dimensionality reduction and in its linear version it consists in the projection of the data on the directions of highest variance, namely the principal components. A non-linear version of the same procedure, known as kernel principal component analysis (KPCA), has been also proposed in [95] where the projection is performed in a (possibly) high dimensional feature space hence enabling to exploit nonlinearity of the data. The free parameter in the algorithm is the number of components to keep and a fundamental question is then if it exists an "optimal" number of components for a given task, or if it is preferrable to keep all of them.

This last question naturally leads to another question that is *how to measure* how effective is KPCA. If the reconstruction error is used as a criterion, recent results ([99]; [123]) suggest that no such an optimal choice exists and the more components we keep the better. On the other hand it is worth noting that one of the main uses of KPCA is as a preprocessing for supervised learning algorithms. In this case one might expect the dimensionality reduction step to influence also the generalization performance since some information is discarded. Going further one might ask if, after KPCA, any kind of regularization is needed at all since again some shrinking of the available information already occurred (see [95] and the discussion in [19]). These issues have been recently addressed in [19] where an algorithm, called kernel projection machine, was proposed which essentially amounts to a KPCA step and then empirical risk minimization with hinge loss function on the projected data. Note that empirical risk minimization is unpenalized and the only free-parameter is the number of components in the dimensionality reduction step. In this case the authors empirically show that indeed an optimal number of components exists when we look at how the generalization performance depends on the dimensionality reduction procedure. As a byproduct they also argued that using some further regularization, for example support vector machines, after KPCA is somewhat redundant and not really necessary. The main goal of this section is to give a proof of such empirical evidences.

In Subsection 1 we show that performing dimensionality reduction, in particular kernel PCA, and then ordinary least squares on the projected data, a procedure known as kernel principal component regression (KPCR), is mathematically equivalent to truncated singular value decomposition or spectral cut-off regularization, which is possibly the most famous regularization scheme for linear ill-posed problems. In this contex the regularization parameter is identified with the number of principal components to keep, m. In Subsection 2, using probabilistic estimates for integral operators based on vector valued law of large numbers we can prove error estimates for KPCR and propose a parameter choice procedure allowing to prove consistency of the algorithm. Indeed such error estimates are made by two error terms, sample and approximation errors, and the best choice for m is the one balancing out the two terms. In Subection 3 we report numerical experiments that confirm theoretical results. We anticipate that in this subsection we will consider data-driven model selection via cross validation, rather than the optimal parameter choice provided by the theory. In fact we note that, even though the bound we obtain conveys the correct qualitative behavior of the error w.r.t. the number of components, and can be shown to be essentially optimal under the given assumptions, nonetheless, since the bound is basically distribution independent in typical applications it will be too pessimistic.

3.2.1 Principal Component Regression and Spectral Cut-Off Regularization

In this subsection we show the equivalence between principal component regression [92] and the regularization algorithm known as spectral cut-off or truncated singular value decomposition (TSVD) [44]. First, we briefly recall the principal component regression algorithm, or rather its *kernel* version. Second we review TSVD regularization. Third we discuss a straightforward connection between the two.

We previously noticed that under our assumptions the covariance operator in the feature space is known to be positive and self-adjoint. In particular we let $(\sigma_i, v_i)_{i\in I}$ be the associated eigensystem[1]. We will assume throughout the data to be centered in the feature space so that the v_i's are the principal components.

Remark 1. *When the data are not centered we cannot asses the equivalence between principal component regression and truncated singular value decomposition unless we consider a modified kernel which corresponds to the features covariance operator*

$$T_{\mathbf{x}} \rightarrow \hat{T}_{\mathbf{x}} = (I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n)T_{\mathbf{x}}(I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n).$$

Spectral cut-off on the non recentered kernel is still an efficient algorithm but it is not evident its connection with principal component analysis because the eigenvectors of $T_{\mathbf{x}}$ and $\hat{T}_{\mathbf{x}}$ may be different.

Again we note that from the computational point of view rather than working with $T_{\mathbf{x}}$ one usually considers the kernel matrix since it can be shown that they share the same spectrum and their eigenfunctions/eigenvectors are related. For theoretical purposes it is convenient to consider simply $T_{\mathbf{x}}$.

Kernel Principal component regression can be seen as a two steps algorithm: the first step amounts to an unsupervised dimensionality reduction via (kernel) principal component analysis; the second step is simply ERM on the projected data. As it is often done in practice we control the projection of the data choosing a threshold λ on the magnitude of the eigenvalues. In other words we only keep $m = m(\lambda)$ components corresponding to eigenvalues bigger than λ. We will show in the following, that such a threshold plays the role of regularization parameter controlling the complexity of the KPCR solution. More in details KPCR can be described in the following steps:

1. decomposition of $T_{\mathbf{x}}$ to obtain $(\sigma_i; v_i)$;

[1] We always assume the eigenvalues to be arranged in decreasing order.

2. projection of the data on the first m components such that $\sigma_m > \lambda$ for fixed $\lambda > 0$:

$$\Phi(x) \to \vec{\varphi}^m(x) = \sum_{j=1}^{m} \langle \Phi(x), v_j \rangle \, \vec{e_j}$$

   where $\vec{\varphi}^m(x) \in \mathbb{R}^m$ and $(\vec{e_j})_j$ is a canonical basis in $\mathbb{R}^m$;

3. ERM:

$$\min_{\vec{w} \in \mathbb{R}^m} \frac{1}{n} \sum_{i=1}^{n} (y_i - \vec{w} \cdot \vec{\varphi}^m(x_i))^2$$

   whose solution is given by $\vec{w} \in \mathbb{R}^m$

$$\vec{w} = \sum_{j=1}^{m} ([(\hat{\varphi}^m)^{\,T} \hat{\varphi}^m]^{\dagger} (\hat{\varphi}^m)^{\,T} \mathbf{y})_j \, \vec{e_j} = \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{y_i}{\sigma_j} \langle \Phi(x_i), v_j \rangle_{\mathcal{H}} \, \vec{e_j}$$

   where $[\hat{\varphi}^m]_{ij} = \vec{\varphi}_j^m(x_i)$ and $[(\hat{\varphi}^m)^{\,T} \hat{\varphi}^m]_{ij} = \sigma_i \delta_{ij}$ (by the definition of (σ_i, v_i)).

The KPCR solution can then be written as

$$f_{\mathbf{z}}^{(PCR)}(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{y_i}{\sigma_j} \langle \Phi(x_i), v_j \rangle_{\mathcal{H}} \langle \Phi(x), v_j \rangle_{\mathcal{H}} \, .$$

We emphasize that in this case the solution $\vec{w}$ is an m dimensional vector.

To describe the **spectral cut-off regularization** it is convenient to remember that from the formulation of ERM in the feature space we can rewrite the solution (3.7) on the spectrum of $T_{\mathbf{x}}$ to get

$$w = \sum_{j=1}^{\infty} \sum_{i=1}^{n} \frac{y_i}{\sigma_j} \langle \Phi(x_i), v_j \rangle_{\mathcal{H}} \, v_j.$$

The above problem is possibly ill-posed [44] and the TSVD regularization simply cuts-off unstable components, that is only $m = m(\lambda)$ components are kept corresponding to eigenvalues bigger than λ. This way we get $w^m \in \mathcal{H}$ such that

$$w^m = \sum_{j=1}^{m} \sum_{=i}^{n} \frac{y_i}{\sigma_j} \langle \Phi(x_i), v_j \rangle_{\mathcal{H}} \, v_j. \tag{3.8}$$

We emphasize that in this case w^m is a function in a possibly infinite dimensional space. The solution can then be written as

$$f_{\mathbf{z}}^{(TSVD)}(x) = \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{y_i}{\sigma_j} \langle \Phi(x_i), v_j \rangle_{\mathcal{H}} \langle \Phi(x), v_j \rangle_{\mathcal{H}}$$

which shows that the solution of principal component regression and spectral cut-off are pointwise equal. The theory of RKH spaces ensures that the obtained solutions are identical, in fact for any $g, f \in \mathcal{H}$ the reproducing property (3.4) ensures

$$f(x) = g(x) \;\; \forall \;\; x \quad \Leftrightarrow \quad \langle f - g, K_x \rangle_{\mathcal{H}} = 0 \;\; \forall \;\; x$$

and this implies that f and g are the same function.

3.2.2 Consistency of Dimensionality Reduction

In this section we prove that if $f_\rho \in \mathcal{H}$ we can derive error estimates of the form (3.3) as well as consistency (and Bayes consistency) of KPCR. Alternatively one should replace f_ρ with the best in the model $f_{\mathcal{H}} = \min_{f \in \mathcal{H}} \mathcal{E}(f)$ (see also [12]). To this aim we note that the parameter we have to choose is the threshold λ on the eigenvalues so that it is convenient to use the notation $f_{\mathbf{z}}^{\lambda}$ in place of $f_{\mathbf{z}}^{(PCR)}$.

Theorem 1. *We let $n \in \mathbb{N}$ and $0 < \eta \leq 1$. Moreover we assume that $f_\rho \in \mathcal{H}$ and $\|f_\rho\|_{\mathcal{H}} \leq R$. Then with probability at least $1 - \eta$ we have*

$$\mathcal{E}(f_{\mathbf{z}}^{\lambda_n}) - \mathcal{E}(f_\rho) \leq \frac{16\sqrt{2}}{\sqrt{n}}(\kappa^2 R^2 + (M+R)^2) \log \frac{4}{\eta} \tag{3.9}$$

where we choose

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log \frac{4}{\eta}.$$

We give the proof in the next section and add some comments. As we previously mentioned, an important consequence of theorem 1 is the existence of an optimal value m_n for the number of principal components which depends on the size of the training set and corresponds to the optimal choice for the parameter λ, that is $m_n = m(\lambda_n)$. At first sight this may appear in contrast with the results in [123], where the authors discuss the behavior of the true reconstruction error which should decrease with the number of dimensions D, the parameter m in our conventions. The reason for this apparent contrast is due to the fact that the reconstruction error quantifies the effect of PCA in an unsupervised setting whereas our bound is on the expected error of a supervised problem. Indeed in a supervised setting if we keep too few components we are over-smoothing whereas if we add too many of them we risk to incur into overfitting thus spoiling the generalization performance.

This result may look similar to the error bound presented in [78] and recalled in [19] where the authors investigate the effect of regularization performed by (kernel) PCA through dimensionality reduction. However it can be noted that such result deals with Gaussian white noise regression in a fixed design setting, whereas we consider random design. Moreover, a related analysis can be found in [118] who considers empirical risk minimization in a reproducing kernel Hilbert space. Indeed the results in such paper show that the number of principal components controls the performance of the algorithm yet the subject of model selection via sample/approximation trade-off is not considered.

Finally as a direct consequence Theorem 1 leads to weak consistency for spectral cut-off regularization in regression

$$\lim_{n\to\infty} \Pr\left(\mathcal{E}(f_{\mathbf{z}}^{\lambda_n}) - \mathcal{E}(f_\rho) \geq \epsilon \right) = 0 \qquad \forall \epsilon > 0$$

and in classification

$$\lim_{n\to\infty} \Pr\left(R(f_{\mathbf{z}}^{\lambda_n}) - R^* \geq \epsilon \right) = 0 \qquad \forall \epsilon > 0$$

where we used (3.3).

3.2.3 Proof of the Error Estimates

In this section we give the proof of the main results. As a main mathematical tool we use estimates of integral operators based on vector valued law of large numbers, to derive the probabilistic error estimates which are the keys to understand the role of the parameter λ. Indeed such error estimates are made by two error terms, sample and approximation errors, and the best choice for λ is the one balancing out the two terms. We follow the same approach as in [12] but the proofs adapted to our setting are considerably simplified. Results of a similar flavor can also be found in [78] for the case of regression with Gaussian white noise and fixed design (see also [19]).

We previously need some notation and facts. First we note that, comparing the ERM solution (3.7) with (3.8), we can rewrite the solution of KPCR as

$$f_{\mathbf{z}}^{\lambda} = f_{\rho}(T_{\mathbf{x}})h_{\mathbf{z}}$$

where f_ρ can be seen via spectral theory as a function on the spectrum of $T_{\mathbf{x}}$ such that $f_\rho(\sigma) = \frac{1}{\sigma}$ if $\sigma \geq \lambda$ and 0 otherwise. Second, we denote with

$$T := \int_X \langle \cdot, \Phi(x)\rangle \, \Phi(x) d\rho_X(x) = \mathbf{E}[T_{\mathbf{x}}]$$

the expected covariance operator and we also denote with

$$h = T_{\mathbf{x}} f_\rho. \tag{3.10}$$

Third we recall the following lemma from [25].

Lemma 1. *Let* $\kappa = \sup_{x\in X} \|K_x\|_{\mathcal{H}}$, $\|f_\rho\|_{\mathcal{H}} \leq R$ *and* $y \in [-M, M]$. *For* $0 < \eta \leq 1$ *and* $n \in \mathbb{N}$ *let*

$$G_\eta = \{\mathbf{z} \in (X \times Y)^n : \|h - h_{\mathbf{z}}\|_{\mathcal{H}} \leq \delta_1, \quad \|T - T_{\mathbf{x}}\| \leq \delta_2\},$$

with

$$\begin{aligned} \delta_1 := \delta_1(n, \eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa(M + R) \log \frac{4}{\eta} \\ \delta_2 := \delta_2(n, \eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log \frac{4}{\eta}. \end{aligned}$$

then

$$\Pr(G_\eta) \geq 1 - \eta.$$

Recalling [33] that we have

$$\mathcal{E}(f) - \mathcal{E}(f_\rho) = \left\| \sqrt{T}(f - f_\rho) \right\|_{\mathcal{H}}^2 \tag{3.11}$$

for all $f \in \mathcal{H}$, in order to prove theorem Theorem 1 we first derive a bound on $\left\| \sqrt{T}(f_{\mathbf{z}}^{\lambda} - f_\rho) \right\|_{\mathcal{H}}^2$ for fixed λ (Theorem 2) and then choose the value $\lambda_n = \lambda(n)$ optimizing the bound

Theorem 2. *We let $n \in \mathbb{N}$ and $0 < \eta \leq 1$. We assume that $\lambda < 1$ and*

$$\lambda \geq \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log \frac{4}{\eta}. \tag{3.12}$$

Moreover we assume that $f_\rho \in \mathcal{H}$ and $\|f_\rho\|_{\mathcal{H}} \leq R$. Then with probability at least $1 - \eta$ we have

$$\mathcal{E}(f_{\mathbf{z}}^\lambda) - \mathcal{E}(f_\rho) \leq 8(\lambda R^2 + \frac{C}{\lambda n}) \tag{3.13}$$

where $C = C(\eta, \kappa, M, R) = 8\kappa^2(M + R)^2(\log \frac{4}{\eta})^2$ does not depend on λ and n.

Proof of Theorem 2. In this proof we use the inequalities in the above lemma which holds with probability at least $1 - \eta$ with $0 < \eta \leq 1$. Recalling (3.11), we consider the following error decomposition

$$\left\| \sqrt{T}(f_{\mathbf{z}}^\lambda - f_\rho) \right\|_{\mathcal{H}}^2 \leq 2 \left\| \sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda) \right\|_{\mathcal{H}}^2 + 2 \left\| \sqrt{T}(f^\lambda - f_\rho) \right\|_{\mathcal{H}}^2 \tag{3.14}$$

where

$$f^\lambda = f_\rho(T_{\mathbf{x}})h \quad \text{with } h \text{ given by } (3.10)\ .$$

We now separately bound the two terms in the right-hand side. The first term can be decomposed as

$$\sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda) = \sqrt{T} f_\rho(T_{\mathbf{x}})(h_{\mathbf{z}} - h) = \sqrt{T_{\mathbf{x}}} f_\rho(T_{\mathbf{x}})(h_{\mathbf{z}} - h) + (\sqrt{T} - \sqrt{T_{\mathbf{x}}}) f_\rho(T_{\mathbf{x}})(h_{\mathbf{z}} - h). \tag{3.15}$$

We note that the inequality

$$\left\| \sqrt{T} - \sqrt{T_{\mathbf{x}}} \right\| \leq \sqrt{\|T - T_{\mathbf{x}}\|} \leq \sqrt{\delta_2} \leq \sqrt{\lambda} \tag{3.16}$$

follows from Theorem 8.1 in [79], Lemma 1 and Assunption (3.12).
Moreover from the definition of operator norm and standard results of spectral theory

$$\|g(\mathrm{A})\| = \sup_{\sigma \in \Lambda(\mathrm{A})} g(\sigma) \tag{3.17}$$

where $\Lambda(\mathrm{A})$ is the set of the eigenvalues of the operator $\mathrm{A} : \mathcal{H} \to \mathcal{H}$, it is easy to see that

$$\|f_\rho(T_{\mathbf{x}})\| \leq \frac{1}{\lambda}$$

and

$$\left\| \sqrt{T_{\mathbf{x}}} f_\rho(T_{\mathbf{x}}) \right\| \leq \frac{1}{\sqrt{\lambda}}.$$

If we now take the norm in (3.15) we get

$$\left\| \sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda) \right\|_{\mathcal{H}} \leq \frac{2}{\sqrt{\lambda}} \|h_{\mathbf{z}} - h\|_{\mathcal{H}} \leq \frac{2}{\sqrt{\lambda}} \delta_1. \tag{3.18}$$

We now deal with the second term in the r.h.s. of (3.14). We can write

$$\begin{aligned} \sqrt{T}(f^\lambda - f_\rho) &= \sqrt{T}(I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho \\ &= \sqrt{T_{\mathbf{x}}}(I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho + \\ &\quad +(\sqrt{T} - \sqrt{T_{\mathbf{x}}})(I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho. \end{aligned} \tag{3.19}$$

We can bound this term recalling that by assumption $\|f_\rho\|_{\mathcal{H}} \leq R$ and noting that definition (3.17) implies

$$\|I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}}\| \leq 1 \quad \text{and} \quad \left\|(I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}})\sqrt{T_{\mathbf{x}}}\right\| \leq \sqrt{\lambda}.$$

We note that operator $f_\rho(T_{\mathbf{x}})T_{\mathbf{x}}$ is exactly the projection operator on the subspace spanned by the eigenvectors of $T_{\mathbf{x}}$ with eigenvalue greater or equal to λ, whereas $I - f_\rho(T_{\mathbf{x}})T_{\mathbf{x}}$ is the projection operator on the orthogonal subspace. We can now take the norm of (3.19) and use (3.16) to get

$$\left\|\sqrt{T}(f^\lambda - f_\rho)\right\|_{\mathcal{H}} \leq 2\sqrt{\lambda}R. \tag{3.20}$$

The estimate in (3.13) follows plugging (3.20) and (3.18) into (3.14) and using the definition of δ_1. □

We are now ready to give the proof of Theorem 1.

Proof of Theorem 1. The proof of the theorem is straightforward. In fact since the sample error increases with λ while the approximation error decreases, in order to get the best error we should take the value of λ which gives a good trade-off between the two terms. To this end we set the two terms to be of the same order

$$\lambda_n = \frac{1}{\lambda_n n} \quad \Rightarrow \quad \lambda_n = O(\frac{1}{\sqrt{n}}).$$

Then, in order to be consistent with condition (3.12), we can choose the following value for λ_n

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}.$$

Substituting λ_n in (3.13) we obtain the rate (3.9). □

3.2.4 Numerical Experiments

In this section we present some numerical results to illustrate the behavior of principal component regression on real and simulated data.

The real data experiments have been carried out on two datasets available at `http://www.ics.uci.edu/~mlearn/MLSummary.html`. In the first one we analyzed the Wisconsin diagnostic breast cancer database on benign vs malignant classification. The dataset is made of $n = 569$ examples divided in two classes and described by $d = 30$ features. In the second experiment we examined the SPECTF heart database. This dataset is made of $n = 267$ instances (patients) and $d = 44$ attributes per instance. Each of the patients is classified into two categories: normal and abnormal.

In both experiments we first partitioned the dataset in two balanced subsets, training and test set. As for the parameter choice, despite optimality of the bound, in practice it is going to be too pessimistic to be used with few examples, being basically distribution independent. Indeed

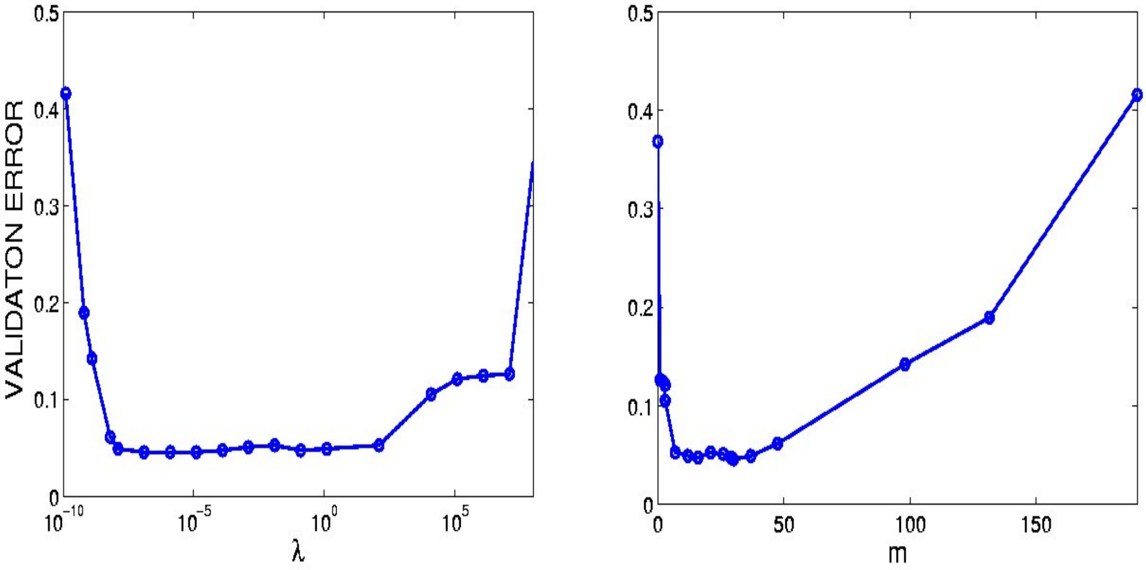


Figure 3.1: 5-fold cross validation error vs λ and m for the brain cancer dataset

the theoretical results of Subsection 3.2.2 highlight the regularization role of the number of dimensions but yet in practice we often need some data driven procedure, such as cross-validation, to choose it. Therefore we determined an optimal value for the regularization parameter via 5-fold cross validation on the training set. For each value of the parameter we estimated the average misclassification error and the median number of principal components that survived the thresholding. Hence we obtained a curve describing an estimate of the expected error as a function of either the regularization parameter λ or the corresponding number of selected components m (see Figure 3.1 and 3.2), choosing the optimal value of the parameter λ_n and m_n as the minimum of such curve. Finally we run the algorithm on the entire training set with the value for λ_n provided by the 5-fold cross validation, and computed the misclassification error on the test data. In order to obtain a more precise estimate of the test error we repeated the entire protocol for 50 different splits of the total dataset in training and test set and averaged the results on these repetitions.

Comparisons with the original results from these two data sets show a lower prediction accuracy (96% against 97.5%) for the breast cancer data set and a higher prediction accuracy (80% against 77%) for the SPECTF data set. However the main purpose of these experiments has been to empirically demonstrate the possibility of choosing an optimal value for the number of components rather than searching for an accurate predictor. Indeed, Figure 3.1 and 3.2 clearly indicates the existence of an optimal value for the threshold which corresponds to an optimal number of principal components to be used in the determination of the classifier. Taking into account more than m_n components can only increase the prediction error.

We also investigated the effect of spectral cut-off on a toy example based on a Gaussian linear regression model $y = \beta x + \epsilon$, where $x \in \mathbb{R}^d$ and $d = 40$. We run the algorithm on training sets of

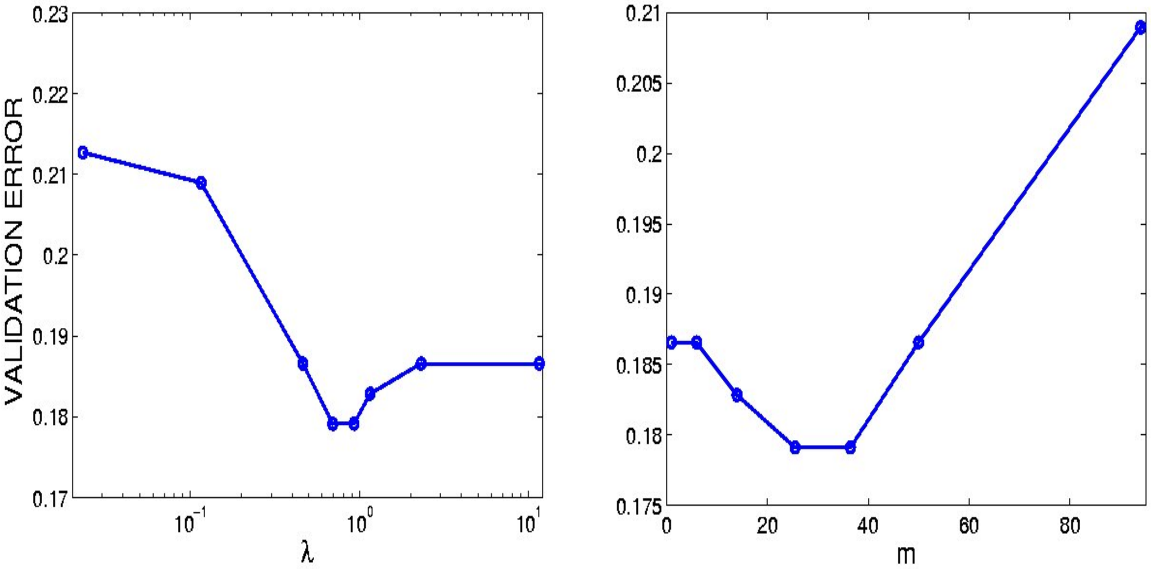


Figure 3.2: 5-fold cross validation error vs λ and m for the SPECTF heart dataset

increasing number of samples with different values of the parameter λ and evaluated the error on a test set of 5000 instances. As expected Figure 4.2 clearly shows that m_n, corresponding to the minimum of the test error for different n, reaches the maximum number of components only for large data sets, whereas a limited number of training instances is better generalized with a limited number of principal components. In order to better understand the effect of further regularization after Principal Component Analysis, we evaluated the test error committed by regularized least squares(RLS) on the first m_{opt} principal components. From Figure 4.2 we can see that RLS do not improve prediction performance since the test error is always approximately equal or greater than the error committed with just spectral cut-off ($\lambda = 0$). This result confirms that regularization has already been taken into account during the preprocessing step.

3.2.4.0.1 Computational comments We also observe that even though in principal component regression the empirical risk minimization algorithm deals with shorter vectors, that is the m-dimensional projection of the data, most of the computation is performed in the preprocessing step which projects the data on the m principal component; therefore the benefit of dealing with smaller matrices is paid with the drawback of the computationally demanding projection. On the other hand, truncated singular value decomposition deals with possibly infinite dimensional vectors, but all the computation is confined to the construction and diagonalization of the covariance matrix or its dual kernel matrix.

Remark 2. *Though we just considered the supervised case the conclusions we draw can be of interest in the context of semi-supervised learning since recently proposed techniques are based on the use of the principal components of data driven kernel for function approximation ([13]; [28]). The extension of our analysis in the case where unlabeled data are available is an interesting direction for future work.*

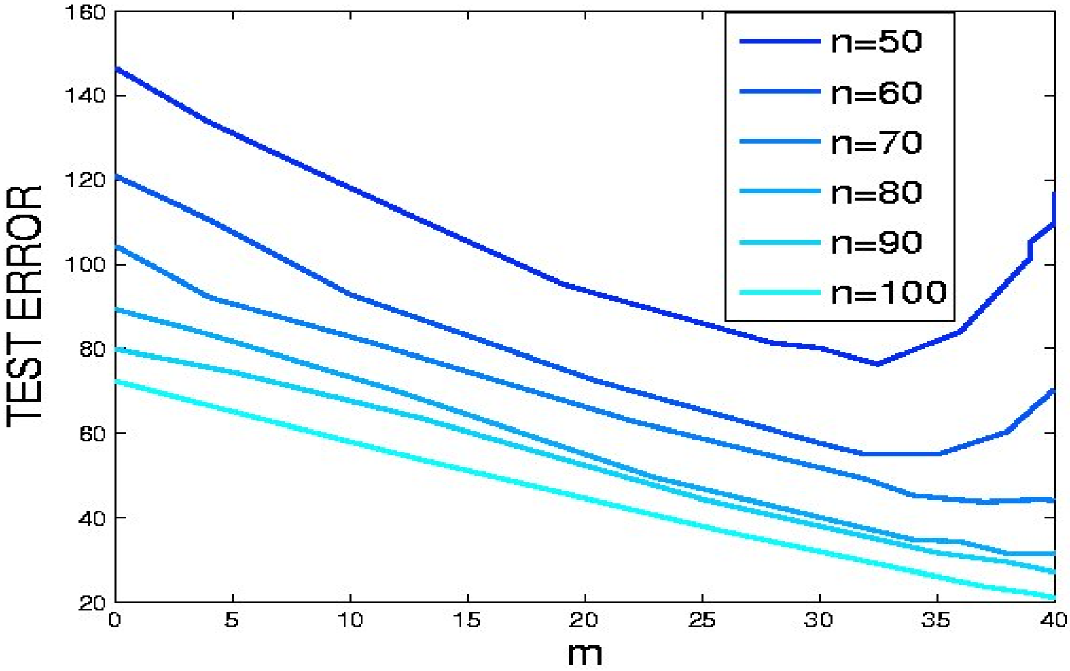


Figure 3.3: Test error vs λ of PCA + ERM for the toy example

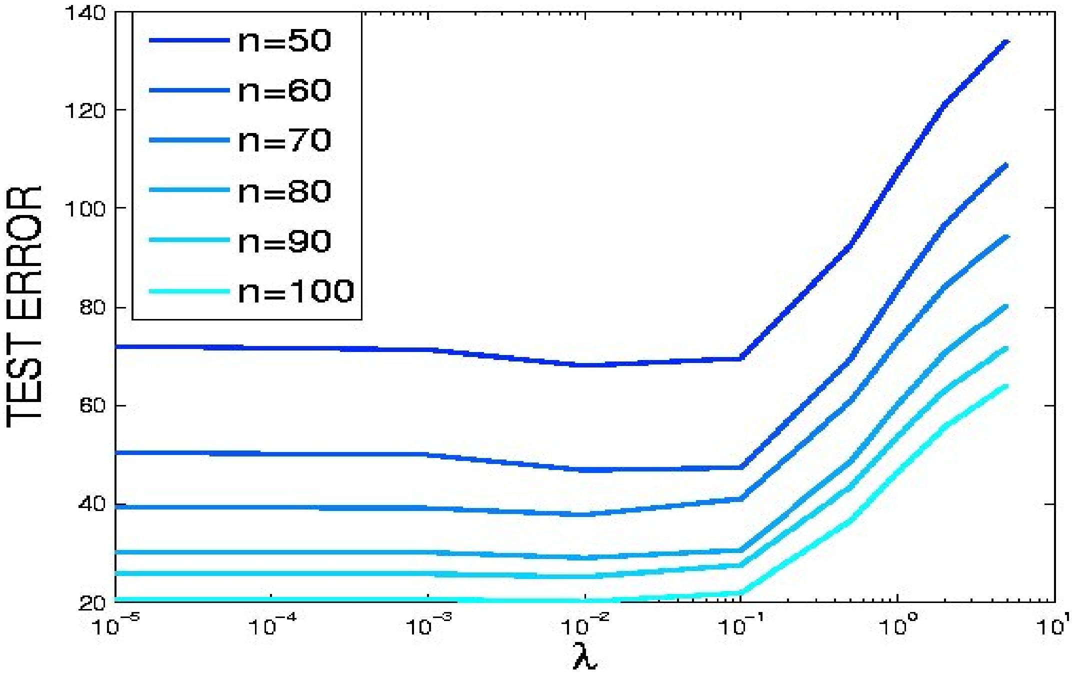


Figure 3.4: Test error vs λ of PCA + RLS for the toy example

3.3 Regularization Properties of Feature Selection

In the previous section we have shown that dimensionality reduction can be regarded as a regularization step that allows to improve prediction accuracy of a subsequent learning phase on the reduced data. Indeed (unpenalized) empirical risk minimization on the projected data does not incur in overfitting if the projection step is suitably tuned, and thus does not need any further regularization.

In fact, we have shown that, when Principal Component Analyis is used as a preprocessing to a supervised learning task, model selection via sample/approximation trade-off induces an optimal value λ_n for the threshold on the eigenvalues and hence a corresponding optimal number of dimensions $m_n = m(\lambda_n)$ to be used in the projection step. This result apparently goes against the intuition that adding more dimensions, and therefore more information from the distribution, the result should improve. Indeed such an intuition is misleading when the data are finitely sampled from a probability distribution; in fact, from (3.13), we can observe that, when m increase (that is λ decreases), the approximation error decreases, but the sample error increases. Therefore an optimal number of dimensions, m_n, exists which depends on the size of the training set and such that using more than m_n dimensions will cause a decrease in the predicting power of the solution.

The theoretical underpinning for sparse regularization, which will be analyzed in the next chapter, is based on similar considerations adapted to the specific case of feature selection. Indeed feature selection can be regarded as a special case of dimensionality reduction based on a stronger prior knowledge, that identifies the features as the dimensions to be reduced. Moreover, whilst in Principal Componnet Regression, regularization is guided by the a priori intuition that smooth solutions are preferrable, regularization by discarding some noisy features can be motivated by the a priori knowledge that some features are more informative than others.

The intuition that an excessive amuont of information could reduce prediction performance was already known since the 70's for classification problems for two-Gaussian distribuited classes. In this context the problem is known under the name of "peaking phenomenon" and was already demonstrated almost four decades ago in [64, 108], and reconsidered more recently in [90, 86], where the optimal number of features was analyzed as a function of the training set size. These works show that if the true statistical model is known, or if the training set is unlimited, any additional feature can only improve accuracy. However, when the sample is finite, additional features can degrade the performance of many classifiers, even when all the feature are statistically independent and carry information on the output.

Despite their conceptual importance, works of this flavor lack generality, due to the restrictive conditions required for their theoretical resulst to hold. Recently a fast expanding literature focused on the need of feature selection in supervised regression and classification tasks with low size training sets, in particular in the area of compressed sensing. In this field feature selection via ℓ^1 regularization is used to reconstruct undersampled signals, apparently violating Shannon/Nyquist sampling theorem. In fact, the central idea in compressive sensing is that the number of samples needed to capture a signal depends primarily on its structural content, rather than on its bandwidth. In the statistical learning setting, the papers [11, 22, 67] discuss optimality of the ℓ^1-norm minimization in a minimax setting. Such results hold assuming that the dictionary is finite (possibly depending on the number of examples) and satisfies some assumptions about the linear independence of the relevant features. Finally a theoretical foreground with consistency proof has been provided in [30] for ℓ^1-ℓ^2 regularization.

Given these premises, in the next chapter we will show how to exploit sparse regularization to perform variable selection and how to adapt it to different kinds of prior knowledge.

Chapter 4

Selecting nested lists of relevant variables

In the last decade a great amount of supervised learning techniques have been proposed to address the problem of feature selection, and many of them proved to be quite efficient in controlled settings. Nonetheless, while accurate prediction can be achieved by means of many different techniques, identification of the relevant variables, due to intra-variable correlation and the limited number of available samples, is a much more elusive problem. Small changes in the measurements often produce different features subsets, and solutions which are both sparse and stable are difficult to obtain. In this chapter we propose an innovative two-stage sparse regularization method able to learn linear models characterized by a high prediction performance. Such a method has been introduced in [34] and later refined in [31] and [9]. The selection core is based on ℓ_1-ℓ_2 regularization, introduced by Zou and Hastie in [122], followed by pure ℓ_2 regularization, the latter being introduced in order to overcome the bias induced by the over-shrinkage phenomenon characterizing sparse regularization. prediction performance. The main contribution of the work presented in this chapter is to have inserted the otherwise non very efficient ℓ_1-ℓ_2 regularization in a two-stage protocol, based on a double optimization, which is able to learn sparse linear models characterized by a high prediction performance and extremely good stability of the selected features. Moreover, by varying a correlation parameter these linear models allow to trade sparsity for the inclusion of correlated features and to produce variable subsets which are almost perfectly nested. Experimental results on synthetic data confirm the interesting properties of the proposed embedded method for feature selection.

In Section 1, wer present a brief survey of a specific class of embedded methods for feature selection, known as sparse regularization techniques. In this context we examine ℓ^1 type penalties, that give rise to well-known sparsity based techniques such as, Lasso, Elastic, Net, Group Lasso and Sparse Multi Kernel Learning.

In Section 2 we investigate the effect of a second ℓ^2-penalized optimization following feature selection, performed by means of sparse regularization. We provide empirical evidence that, when the second step amounts to running a (regularized) least squares optimization on the selected features, prediction performance is effectively increased.

In Section 3, we restrict to ℓ^1-ℓ^2 regularization and describe our two-stage approach: the first stage establishes a minimal subset of features relevant to the classification or regression task under investigation; the second stage produces a one-parameter family of variables subsets, showing a remarkable nesting property and similar performance in terms of classification/prediction tasks.

In Section 4 we design a robust procedure for performing the model assessment and selection separately from the training step, by means of cross-validation. We described how such procedure applies to our two-stage selection method, and illustrate how to extend it to the analysis of particularly small data sets.

In Section 5 we summarize the details of the selection core, that is the algorithm for performing variable selection. We describe the damped thresholded Landweber iterative algorithm we used for computing the ℓ_1 regularization solution, and the stopping rule we employed. Moreover we illustrate an acceleration that exploits some heuristic to reduce the computing time by a factor of about 100.

Finally, in Section 6 we challenge our system on synthetic data, where we can compare the algorithm results with the true model.

4.1 Sparse Regularization

In standard learning algorithms regularization is achieved by searching for a solution minimizing a term depending on the data penalized by one or more terms enforcing regularity in the solution. The data term is the empirical risk, $\mathcal{E}_{\mathbf{z}}(f) = \frac{1}{n}\sum_{i=1}^{n} l(f(x_i), y_i)$, i.e. the empirical counterpart of the expected risk $\mathcal{E}(f)$ measuring the loss of function $f(x)$ on the training data $(x_i, y_i), i = 1, \ldots, n$. The minimization of the empirical risk (ERM)

$$\operatorname*{argmin}_{f \in \mathcal{H}} \mathcal{E}_{\mathbf{z}}(f), \tag{4.1}$$

is known to be an ill-posed problem, depending on the choice of the hypothesis space, $\mathcal{H}$. If the hypothesis space is too large, the solution of (4.1) will incurr in overfitting. A common apporach to to avoid overfitting, i.e. to *regularize* the problem, is to add to the data-misfit term a penalty which allows to control the complexity of the solution The most usual choice is a penalty proportional to the square of the ℓ^2 norm of function f in the hypothesis space $\mathcal{H}$. When the empirical risk is a convex functional of f, strict convexity of the ℓ^2 norm $\|f\|^2_{\mathcal{H}}$ guarantees uniqueness and stability of the solution. By varying the loss function l, we recover the most famous learning algorithms. When the loss is the Hinge loss, the objective function writes

$$\frac{1}{n}\sum_{i=1}^{n}(1 - f(x_i)y_1)_+ \lambda \|f\|^2_{\mathcal{H}},$$

which minimization coincides with *Support Vector Machine* [109] (SVM). Instead, with the quadratic loss, the empirical risk is the squared error, and we recover ℓ^2 regularization,

$$f_{\mathbf{z}} = \operatorname*{argmin}_{f \in \mathcal{H}} \sum_{i=1}^{n} (f(x_i) - y_i)^2 + \lambda \|f\|_{\mathcal{H}}^2$$

also called *ridge regression* [62, 59], regularized least squares [44, 17], or regularization networks [89].

When dealing with feature selection problems, the main idea is to interpret sparsity as regularity and therefore employ a penalty term which explicitly enforces sparsity of the solution as in (2.3). Let us consider a linear model, modelling the relation between x, which is represented on a finite dictionary of d features ψ_j, and y as $y = \beta \cdot x$. i.e. $y \sim f(x) = \sum_{j=1}^{d} \beta_j \psi_j(x)$. The goal is to determine a *sparse* model β^*, i.e. a model of cardinality much smaller than d – that is, a vector β^* with only s entries different from zero (with $s << d$) – for which the expected risk, $\mathcal{E}[\beta^*]$, takes on a small value. We recall that the components of the model vector are called regression coefficients or weights. In this context an ℓ^2 type penalty proves to be inadequate, since in spite of its regularization properties, ℓ^2 regularization does not performs variable selection. If we define Ψ the $n \times p$ matrix with $\Psi_{ij} = \psi_j(x_i)$ and Y the $n \times 1$ vector with $Y_i = y_i$, the penalized least squares linear regression is the solution to the optimization problem

$$\operatorname*{argmin}_{\beta \in \mathbb{R}^d} \frac{1}{n} \|Y - \Psi\beta\|_{\mathbb{R}^n}^2 + \lambda \sum_{j=1}^{d} \beta_j^2 \tag{4.2}$$

which unique minimizer is a model vector with typically all entries different from zero. The linear computational schemes arising from this framework are easy to implement and produce numerically stable solutions which, for optimal values of the regularization parameter, lead to accurate predictions. ℓ^2 regularization, however, tends to distribute the weights evenly among correlated features and, thus, is not suited for performing feature selection.

Alternatively, one could replace the ℓ^2-norm by another penalty having the effect of enforcing sparsity of the coefficient vector. In this section we review the most famous optimization principles and algorithms that perform feature selection by means of sparse regularization, starting from an ℓ^0 type penalty, and then focusing on ℓ^1 regularization and its variations.

4.1.1 ℓ^0 regularization

An intuitive choice for $\Omega(\beta)$ is clearly the zero norm of the coefficient vector $\|\beta\|_0$. With this choice, minimization of (2.3) is interpreted as finding a β with as few non zero coordinates as possible such that the derived linear model is consistent on the training set. This approach is affected by two limitations: first, as mentioned before, minimization of (2.3) is known to be NP-hard and hence is not computationally feasible, mainly because of the lack of convexity of the penalty; secondly, the minimization of the risk penalized with the $\ell 0$ norm do not perform regularization, since it is independent of the trade-off parameter la, and is thus at risk of over-fitting. Indeed when the number of training samples is much smaller than the number of dimensions, the chance of having a noisy feature completely correlated with the output target is extremely high. These drawbacks support the idea that exact minimization of the $\ell 0$ norm is not desired and it motivates approximations that can be efficiently computed and that allow to control the generalization error.

4.1.2 ℓ^1 regularization

The most popular alternative to ℓ^0 norm minimization is the ℓ^1 norm minimization, where the number of the non null coefficients is substituited by the sum of the coefficient absolute values, $\|\beta\|_1 = \sum_{j=1}^{d} |\beta_j|$. Even if it appear accompanied by different loss functions, ℓ^1 regularization has its most popular form when the risk is the mean square error. In this case the objective functionis given by

$$\frac{1}{n} \|Y - X\beta\|^2 + \tau \|\beta\|_1 , \tag{4.3}$$

where X the $n \times p$ matrix with X_{ij} the j-th component of x_i and Y the $n \times 1$ vector with $Y_i = y_i$. In ℓ^1 regularization - also called LASSO regression in [106] - overtting is avoided by enforcing sparsity, i.e. by favoring model vectors with only a small number of entries different from zero, and at the same time shrinking the coefficients to zero. Algorithms like LASSO (Least Absolute Shrinkage and Selection Operator) [106], LARS [38] and basis pursuit [27] have been proposed to find the minimizer of the corresponding objective function in different contexts.

4.1.3 ℓ^1-ℓ^2 regularization

In applications where the solution is known to depend on a relatively small number of features, ℓ^1 regularization appears to be quite appropriate. The minimizer is known to be unique (except for very special configurations of the inputs) and stable with respect to noise in the output data y_i. However, small changes in the components of the input data x_i lead to a different feature selection, typically with no appreciable change in the overall expected risk (or accuracy in the performance) of the obtained model. Consequently, when the inputs are affected by noise or the number of examples is small compared to the number of features, the selection of the components of the model vector β might be driven by random fluctuations, in particular in the presence of correlated features, it is not guaranteed that the ℓ^1 penalty captures all relevant features.

To overcome this drawback, [122] have proposed a method, called *(naïve) elastic net*, which makes use of a linear combination of both ℓ^1-norm and ℓ^2-norm penalties to the purpose of selecting sparse groups of correlated features. Indeed, empirical evidence [122] indicates that an additional ℓ^2 norm to the objective functional

$$\frac{1}{n} \|Y - \Psi\beta\|^2 + \mu \|\beta\|_2^2 + \tau \|\beta\|_1 , \tag{4.4}$$

produces stable solutions, exhibits an interesting grouping effect by selecting correlated features (due to the presence of the ℓ^2 penalty term). The balance between the penalty and the data-depending term, as well as the respective weight of the two norms, is tuned by means of two nonnegative parameters, called the *regularization parameters* and multiplying respectively the ℓ^1-norm and the squared ℓ^2-norm.

The above objective functional has been studied from the theoretical point of view in [30]. In context of face recognition it has been studied in [34] +REF, whereas applications to the analysis of gene expression data are present in [31, 9] and in this Ph.D thesis.

4.1.4 Block-wise Feature Selection

Increasingly more popular techniques for feature selection are *Group Lasso* [117], *and Sparse Multiple Kernel Learning*, which have the attractive property of producing a block-wise sparse solution and therefore of selecting specific groups of variables, according to some prior knowledge on the blocks.

Group Lasso
This approach has been proposed as an extension of the lasso when the features are partitioned in p pre-assigned non-overlapping blocks, and selection has to be performed block-wise. Formally, in the Group Lasso objective function the sum of the coefficient absolute values is replaced by sum of the norm of the coefficient vector restricted to the blocks:

$$\frac{1}{n}\|Y - \Psi\beta\|^2 + \sum_{k=1}^{p} \sqrt{\sum_{j \in \mathcal{I}_k} \beta_j^2}$$

where $(\mathcal{I}_k)_{k=1}^{p}$ is a partition of $\{1, \ldots, d\}$.

Sparse Multiple Kernel Learning
Multiple Kernel Learning was introduced in order to provide more flexibility of the solution, and to reflect the fact that typical learning problems often involve multiple, heterogeneous data sources. Without going into details – we will review the properties of kernel function and of Reproducing Kernel Hilbert Spaces in the next chapter –, in Multiple Kernel Learning the solution can be written as sum of function belonging to different hypothesis spaces, $f = \sum_{k=1}^{p} f_k$, with $f_k \in \mathcal{H}_k$, and the goal is to learn the combination of functions that minimzes the objective functional.

In Sparse Multiple Kernel Learning, the objective functional is given by:

$$\mathcal{E}_{\mathbf{z}}(f) + \tau \sum_{k=1}^{p} \|f_k\|_{\mathcal{H}_k}$$

which is some how analogue to Group Lasso in the sense that selection is performed block-wise. In fact, its minimizer will have part of its components set to zero.

4.2 Improving prediction accuracy of sparse regularization with a double optimization

[73] have shown that if the prediction accuracy is used as a criterion to choose the tuning parameter, ℓ^1-norm penalized regression methods (like LASSO) provide consistent estimates in terms of prediction accuracy but not necessarily in terms of variable selection. Indeed, according to empirical findings present in the literature [24], ℓ^1-regularization tends to overshrink the non null entries of the estimated coefficient vector. In this section, we study such over-shrinkaging phenomenon on a simple toy example. Moreover, in order to overcome this negative effect, we propose to substitute pure ℓ^1-regularization with a double optimization: in an initial step certain amount of variables is selected through the minimization of the empirical risk regularized with an ℓ^1-like (and eventually ℓ^2) penalty, in a second step the weights of the selected variables are refined through ℓ^2 regularization. Finally we provide empirical evidence of the fact that very good prediction accuracy and correct variable selection can be both achieved by coupling the ℓ^1-norm penalized regression method with a second optimization step restricted to the selected variables.

Theoretical results (see [120] for details) state that, when the regularization parameter is appropriately tuned, asyntotically (as the sample size n gets large) ℓ^1 regularization selects the true model. However, in practice the optimal value of the parameter is unknown, and parameter selection is thus performed via minimization of the (cross) validation error. This induces a choice for the parameter which is optimal with respect to prediction accuracy, but tends to select too many variables. In [73] the authors have shown that, for fixed d and orthogonal design (i.e. the covariance matrix $\Psi^T\Psi$ is orthogonal), the ℓ^1 regularization estimate that is optimal in terms of parameter estimation does not give consistent model selection. Such phenomen is due to the fact that ℓ^1 regularization underestimate the weights of the relevant variables. As a consequence, in problems where the main goal is variable selection, prediction accuracy based criteria alone are not sufficient for this purpose.

In order to better understand this effect, let us consider a linear regression model, $y = x \cdot \beta^* + \epsilon$, where the regression function f_ρ is a linear combination of a small subset of the input variables, the samples x and the *true* model β^* belong to $\mathbb{R}^{1000}$, $y \in \mathbb{R}$, and ϵ represents the noise. We generate a sparse model by setting all the entries, but the first three, of the *true* model vector β^* equal to zero. More precisely we set $\beta^* = (1, 1, 1, 0, \ldots, 0)$. The training set is built by randomly drawing 50 samples from a uniform distribution between $[-1, 1]$ for each of the 1000 components, while the noise is sampled from a zero-mean Gaussian distribution with standard deviation $\sigma = 0.5$. The training set is thus built by drawing 50 samples from the above distribution. We then perform ℓ^1 regularization on the toy data set, for different values of the parameter τ, and plot the square error and the model error versus the τ in Figure 4.1. We then perform ℓ^1 regularization on the toy data set, hence learning a model $\beta(\tau)$ for different values of the parameter τ and evaluate the *model error*:

$$E(\beta) = \sum_{j=1}^{d} 1(1([\beta^*]_j) - 1(\beta_j)),$$

counting how many entries of β have been "misselected", and the square error:

$$\mathcal{E}(\beta) = \|\beta^* - \beta\|_2^2 \,,$$

measuring the euclidean distance between the learned and true coefficient vectors. Finally we plot the square error and the model error versus the ℓ^1 parameter τ in Figure 4.1. Clearly the minimum reached by the model error does not correspond to the parameter giving minimum square error; furthermore, the former minimum is given by a bigger τ. Indeed ℓ^1 regularization, when tuned to minimize the squared error, misses the right model, in that it selects too many variables.

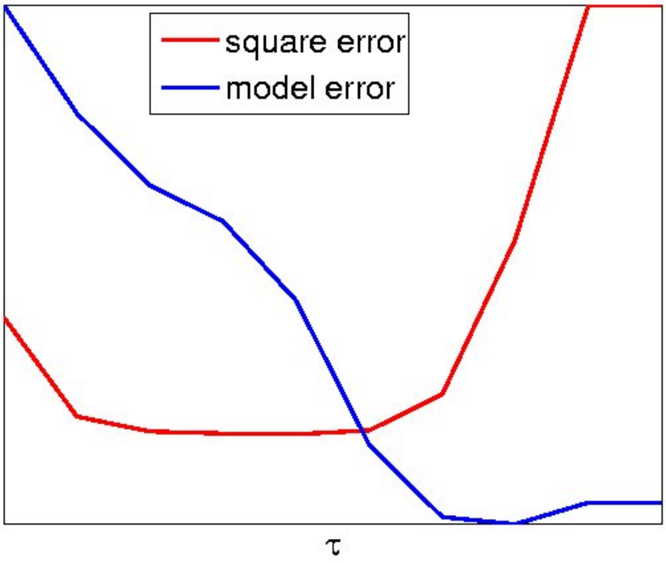


Figure 4.1: Square error and model error of ℓ^1 regularization for the toy example.

In order to overcome the bias induced by the overshrinkage phenomenon, we propose a double optimization procedure where ℓ^1 regularization is used to perform variable selection, whereas ℓ^2 regularization (RLS), or ridge regression, is peformed on the selected variables. This allows us to exploit the ability of ℓ^1 regularization of selecting features – due to the relatively large value we obtain for the corresponding parameter τ – and nevertheless to reach high prediction accuracy through the second minimization step (due to the relatively small value we obtain for the RLS parameter). A similar approach was proposed in [24] applied to the Dantzig selector [23], which corresponding minimization principle is very similar to ℓ^1 regularization. In these works, the authors are aware that the optimal regularization parameter with respect to model selection leads to relatively large bias in estimating the sparse regression coefficients of the true model. To reduce the bias, they therefore suggest a two steps procedure called the Gauss-Dantzig selector which uses the original Dantzig selector for variable selection and then runs ordinary least squares on the selected variables.

At the momoment, a general theory supporting such intuition is still missing. We thus present a toy example to give empirical evidence of how the choice of the optimal value of τ (which effectively drives the feature selection step) is strongly influenced by the presence of a further optimization step restricted to the selected features. The training and validation sets are built by drawing respectively 50, and 1000 samples from the linear regression model described above.

We compare the results obtained with the procedures

(a) ℓ^1 regularization alone, and

(b) ℓ^1 regularization followed by ordinary least squares (OLS) on the selected features,

for different values of τ. The error curves in Figure 4.2 show how procedure (b) allows to reach a lower minimum than (a). The validation error is taken to be the mean-square error. The minimum is clearly reached for a larger value of τ, i.e. when the ℓ^1-norm penalized regression algorithm selects a lower number of features.

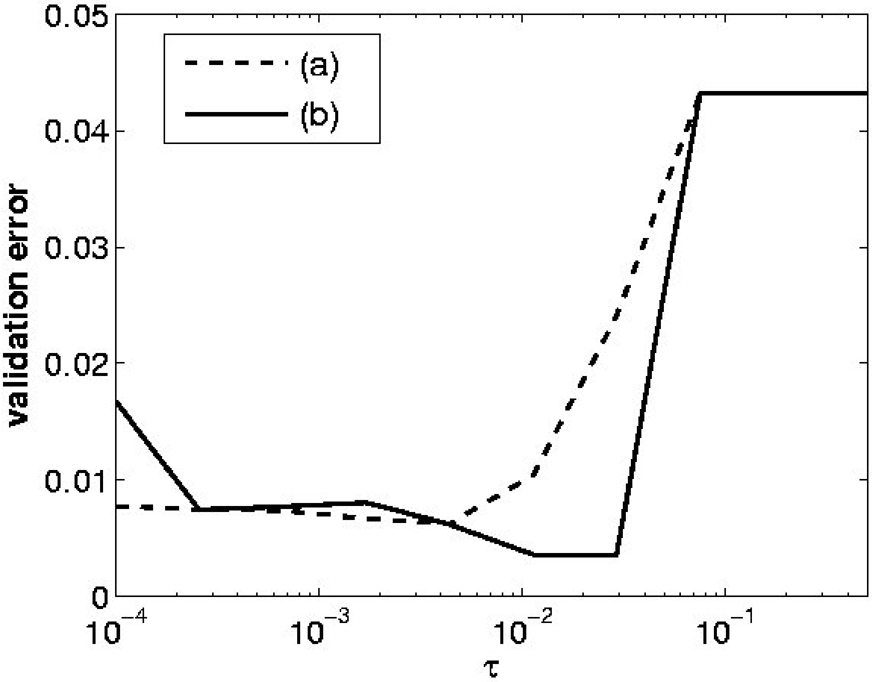


Figure 4.2: Validation error for (a) ℓ^1-reg, and (b) ℓ^1-reg + OLS.

Let us now compare the estimators β^a and β^b obtained, respectively, through (a) with $\tau = \tau^a$ and through (b) with $\tau = \tau^b$ (these parameters minimize the corresponding validation error). In Table 4.1 we report the first three components of $\beta^*, \beta^a, \beta^b$. The last column β^c, represents the output of ℓ^1-norm penalized regression with $\tau = \tau^b$. From Table 4.1, we can see that both β^a and β^b approximate well the relevant components of β^*. However, while β^b correctly selects the model, β^a has many non-zero components besides the first three. This is due to the ℓ^1-norm

β^*	β^a	β^b	β^c
0.6449	0.5667	0.6705	0.3912
0.8180	0.7389	0.8106	0.6388
0.6602	0.5785	0.6794	0.4210

Table 4.1: Comparison of the most relevant components of different regression estimators on a toy example. From left to right: the true weights of the only three non-zero components of the linear model used as a toy example (β^*), the corresponding weights obtained with ℓ^1 regularization (β^a), with ℓ^1 regularization followed by OLS (β^b), and with ℓ^1 regularization with the same τ as β^b (β^c).

penalized regression which, in order to reduce bias, induces an optimal choice of τ smaller than the one needed to correctly identify the model. This can also be seen from the fact that, whereas the ℓ^1-norm penalized regression followed by ordinary least squares selects the correct features (the model β^b has only the first three components different from zero) and returns almost perfect feature weights, the estimator β^c – obtained for $\tau = \tau^b$ – underestimates all three coefficients.

We can thus conclude that the model selection obtained by coupling the two optimization procedures appears to be more effective. especially with very high ℓ_1 parameter, and is thus to be preferred in the presence of highly sparse models. Therefore, in the next section we will apply the same consideration when using ℓ^1-ℓ^2 regularization.

4.3 Two-stage approach for ℓ^1-ℓ^2 regularization

We now focus on the ℓ_1-ℓ_2 optimization problem (4.4) originally presented by [122], and further studied from the statistical point of view in [30]. The above regularization has several interesting properties. First of all, unlike other heuristically motivated methods, it was proved to be a consistent variable selection scheme [30] (as the number of available training samples increases the best possible estimator is eventually reached). Secondly, it takes into account the multivariate effect of many variables together, and avoids discarding correlated variables, in fact, on the one hand the ℓ^1 penalty, enforces selection properties and the ℓ^2 term preserves correlation among input variables, and thus forces the algorithm to simultaneosuly selecting correlated variables. Moreover it has proven to provide successfull solutions to the feature selection problem in many real data experiments, that were originally presented in [34, 31, 9], and that we report in the next chapter of this Ph.D. thesis.

The ℓ_1-ℓ_2 optimization problem was introduced [122] with the name of *(naïve) elastic net* in order to overcome a fundamental drawback of simple ℓ^1 regularization. In fact, as already mentioned in Section 4.1, ℓ^1 regularization has become a popular tool for feature selection in many applications, however, in the presence of correlated features, it is not guaranteed that the ℓ^1 penalty captures all relevant features. In [122] the authors make use of a linear combination of both ℓ^1-norm and ℓ^2-norm penalties to the purpose of selecting sparse groups of correlated features. In fact, adding the ℓ_2 penalty $\mu \sum_{j=1}^{d} |\beta_j|^2$ can be shown to enforce correlated features to have similar weights, so that a whole group of correlated variables is either discarded or selected. The balance between the penalty and the data-depending term, as well as the respective weight of the two norms, is tuned by means of two nonnegative parameters, called the *regularization parameters* and multiplying respectively the ℓ^1-norm and the squared ℓ^2-norm.

Empirical evidence [122] indicates that the naïve elastic net produces stable solutions, exhibits an interesting grouping effect by selecting correlated features (due to the presence of the ℓ^2 norm), but suffers from a quite severe solution bias (due to the shrinkage phenomenon induced by the ℓ^1 regularization term described in the previous section). In fact, good generalization performances are reported only for large values of the μ parameter, case in which the obtained solution is very similar to ridge regression, because, in order to preserve good prediction accuracy, the ℓ^2 penalty tends to dominate so that the method lacks selectivity. In order to contrast bias and enhance the ability of ℓ^1 regularization of promoting sparse solutions, [122] proposed to rescale the coefficients and introduced what they called the *elastic net*:

$$\beta^{(e.n.)} = (1+\mu)\operatorname*{argmin}_{\beta}\{\frac{1}{n}\left\|Y - \tilde{\Psi}\tilde{\beta}\right\|^2 + \tau\left\|\tilde{\beta}\right\|_1 + \mu\left\|\tilde{\beta}\right\|_2^2, \}.$$

Nonetheless, since the scaling factor $(1+\mu)$ depends only on the ℓ^2 parameter, the elastic net takes does amend the effect of the double regularization, but does not cure the shrinking effect that characterizes also pure ℓ^1 regularization.

In light of the empirical results described in the previous section, we start off from the same optimization scheme but explore a different direction, that exploits the effectiveness of the double optimization, and is in line with recent theoretical results [30]. In such a work, in fact, the minimization of (4.4) is shown to yield consistent estimators of the linear model for $n \to \infty$. This means that we can find suitable sequences of parameters $\tau_n, \mu_n = \epsilon\tau_n$, tending to zero as $n \to \infty$, such that we have, in probability,

$$\mathcal{E}[\beta(\mu_n, \tau_n)] \to \ \inf\ \mathcal{E} \qquad \text{for} \qquad n \to \infty.$$

Notice that the above consistency is obtained for any value of the degree of correlation ϵ.

In (4.4) $\epsilon = \frac{\mu}{\tau}$, and we thus employ the parameter $\mu > 0$ as a threshold determining the correlation level above which variables are to be considered belonging to the same group. It is exactly the tuning of the μ parameter that allows us to get an output with more structure than a simple list of variables. Let us explain the two-stage procedure, [31] we use to this aim.

The purpose of **Stage I** is to obtain a solution of minimal cardinality affected by small bias. This result is reached by coupling two optimization procedures and using a very small value of μ with respect to τ. As for ℓ^1 regularization, in our approach to the elastic net, we combine the ℓ^1-ℓ^2 optimization with a subsequent pure ℓ^2 regularization or regularized least squares (RLS) step restricted to the features selected in the elastic-net optimization. In the first optimization, features are selected by minimizing (4.4). In the second one we run a regularized least-squares optimization,

$$\frac{1}{n}\left\|Y - \tilde{\Psi}\tilde{\beta}\right\|^2 + \lambda\left\|\tilde{\beta}\right\|_2^2,$$

where $\tilde{\beta}$ and $\tilde{\Psi}$ represent the weights vector β and the input matrix Ψ restricted to the features selected by the first procedure. The cross-validation protocol which we employ to estimate the optimal values of both τ and λ, and which will be described in details in the next section, yields relatively large values of τ and very small values of λ. Consequently, for the optimal parameter pair, the solution obtained from the first optimization selects a small number of features (characterized by a severe bias) while the regularized least-squares minimization (4.2) restricted to the selected features returns a much more accurate model. Notethat in order to avoid any eventual overfitting, the second optimization amounts to a further regularization. However, in practice the optimal value of the second regularization parameter is extremely small. We can then deduce that the regularization performed during the selection step already guarantees good generalization.

In **Stage II** the objective functions (4.4) and (4.2) are minimized for increasing values of μ, while keeping λ and τ fixed to their optimal values obtained through cross-validation. The resulting one-parameter family of solutions, $\{\beta_\mu | \mu \geq 0\}$, yields lists of relevant features of increasing cardinality. In fact, while β_0 is associated to a minimal set of variables, large values of μ produce lists comprising the basic set and its correlated features, the degree of correlation being controlled by the ratio $\epsilon = \mu/\tau$.

Moreover, due to correlation, we expect that the resulting variables subsets, corresponding to the non-null entries of the one parameter family of coefficients vectors, present a nested structure. Indeed, experiments performed both on synthetic and real data, reported in Section 6.4 and Chapter 7 respectively, show that the obtained lists are almost perfectly nested. In several problems the ability of returning nested list of relevant variables is often regarded as the most precious information for further investigation. This is particularly crucial in biological applications such as gene selection from microarray data, which we will discuss in details in the second part of this Ph.D. thesis.

4.4 Model Selection

When dealing with high-throughput data the choice of a consistent selection algorithm is not sufficient to guarantee good results. It is therefore essential to introduce a robust methodology to select the significant variables not susceptible of selection bias [3] and to use valid statistical indicators to quantify and assess the significance of the results. In both case a robust estimate of the generalization error is needed. In general, the error obtained on an independent data set, i.e. not seen during the learning phase, is used as an estimate of the generalization error.

Ideally, if the data set size is sufficiently large, we can perform hold-out validation and testing, by splitting the samples into training, validation and test set, see Figure 4.3. This allows us to run the algorithm on the training set for different values of the parameter(s), each time evaluating the error committed on the (independent) validation set. Validation error is thus an unbiased estimate of the prediction error and can be minimized to obtain the optimal value of the parameter. Finally a good estimator of the algorithm performance is provided by the error committed on the test set by the solution obtained with the optimal parameter.

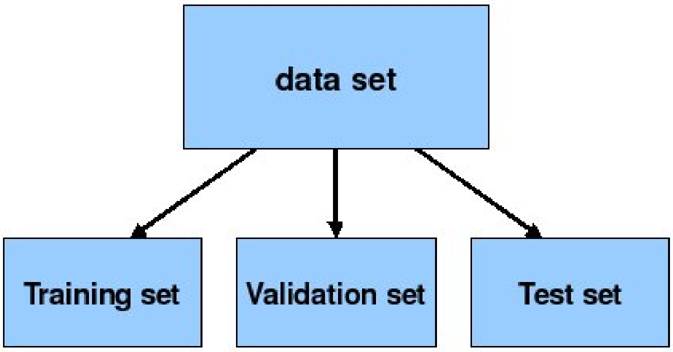


Figure 4.3: Data splitting for large data sets

In most cases, however, the number of available samples does not allow for such a hold-out procedure, and some expediency is needed for guaranteeing unbiased parameter selection. Generalization error is thus estimated by “resampling” the data set; the most common resampling technique are “cross-validation” and “bootstrapping” ([40, 61]). Cross-validation (leave-one-out or k-fold cross-validation) is a form of iterative hold-out testing which repeatedly uses part of the available data to fit the model, and a different part to validate it. On the other hand, in bootstrapping, instead of repeatedly analyzing subsets of the data, one repeatedly analyzes subsamples of the data; each subsample is a random sample with replacement from the full sample. Bootstrapping seems to work better than cross-validation in many cases ([37]). However, the results obtained so far are not very thorough, and it is known that bootstrapping does not work well for some other methodologies such as empirical decision trees ([21, 68]), for which it can be excessively optimistic. Let us focus on cross-validation and show how to apply the two stage procedure described in the previous section in two conditions that differ in data size.

A first, and very common, situation is found when the number of available samples allows only for one hold-out procedure, and the samples are hence partioned into two sets, one will be used for training and validation, while the other will be used for testing; due to the absence of an independent data set for validation, cross-validation is applied to the first set of data to estimate the generalization error, see Figure 4.4. The data set is initially divided in training and test set. The training set is further partitioned in k subsamples $\Psi_1, ..., \Psi_k$ with k depending on the cardinality of the training set. In Stage I, for each subsample Ψ_i, a classifier is first built using as training set the remaining $k-1$ subsamples with τ and λ ranging on a grid in the parameters space, and then validated on Ψ_i. Each classifier is built by minimizing the objective functions (4.4) and (4.2) with the current values of τ and λ and a fixed small value for μ (typically $\mu_0 = 10^{-6}$). For each parameter pair the validation error is estimated as the average error over the k subsamples. Finally the optimal parameter pair, (τ^*, λ^*), is selected as the minimizer of the validation error. In Stage II a family of classifiers is built on the entire training set with $\tau = \tau^*$, $\lambda = \lambda^*$ and for m increasing values of μ. Along with a test error each classifier returns a list of variables indexed by the value of μ. A pseudo-code version of this procedure is summarized in the box below.

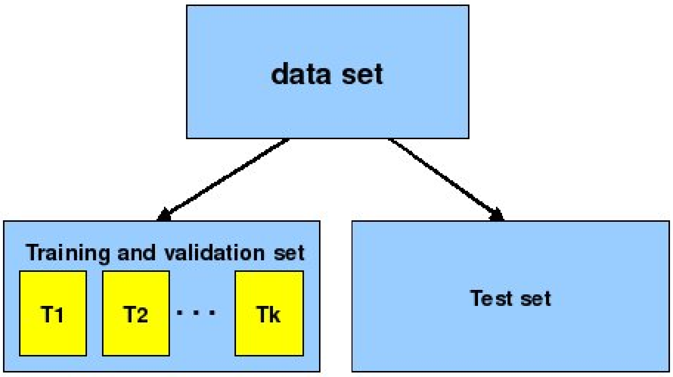


Figure 4.4: Data splitting for intermediate data sets

```
Given
- $(\Psi, Y)$ the training set, and $(\Psi^{test}, Y^{test})$ the test set
- $\{(\Psi_1, Y_1), \ldots, (\Psi_k, Y_k)\}$ partition of $(\Psi, Y)$
- $\mu_0 < \mu_1 < \cdots < \mu_{m-1}$

Stage I
- let $\mu = \mu_0$, $(\tau_t, \lambda_l)_{t \in \mathcal{T}, l \in \mathcal{L}}$ a grid in parameter space
- for $t \in \mathcal{T}$ and $l \in \mathcal{L}$
    for $i = 1$ to $k$ let
        $X_i^{tr} := \Psi_1, ...\Psi_{i-1}, \Psi_{i+1}, ..., \Psi_k$
        $Y_i^{tr} := Y_1, ...Y_{i-1}, Y_{i+1}, ..., Y_k$
        $\beta(t, l, i) :=$ classifier built on $(\Psi_i^{tr}, Y_i^{tr})$ for $\tau = \tau_t, \mu = \mu_0$, and $\lambda = \lambda_l$
        $Err(t, l, i) :=$ error made by $\beta(t, l, i)$ on $(\Psi_i, Y_i)$
    end
    $\overline{Err}(t, l) := \frac{1}{k} \sum_{i=1}^{k} Err(t, l, i)$
end

Stage II
- find $(\tau^*, \lambda^*)$ minimizing $\overline{Err}(t, l)$
- for $i = 0$ to $m - 1$ let
    $\beta_\mu^* :=$ classifier built on $(\Psi, Y)$ for $\tau = \tau^*$, $\mu = \mu_i$, and $\lambda = \lambda^*$
    $Err_i^{test} :=$ error made by $\beta_{\mu_i}^*$ on $(\Psi^{test}, Y^{test})$
end
```

The extreme data size condition is frequently found in the analysis of data coming from modern high-throughput technologies, where, due to resource limitations, most times the available samples do not exceed a few tens, and are thus few compared to the number of variables, typically several thousands. With a small set of training data, a prediction model may not be able to accurately represent the data under analysis. Similarly, a small test set may contribute to an unreliable prediction quality assessment. The problems of building prediction models based on small datasets and the estimation of their predictive quality deserve a more careful consideration. It is therefore crucial to exploit all the data, though avoiding over-fitting. To this aim a splitting cascade procedure is needed in order to select the model (i.e. the optimal parameters), and to assess the statistical significance of the experiment performed [52]. We thus present an unbiased model selection protocol which is able at the same time to identify the most relevant variables and to achieve a good prediction performance even when dealing with very small data sets. We do not report the pseudo-code for this protocol, but sketch it in Figure 4.5.

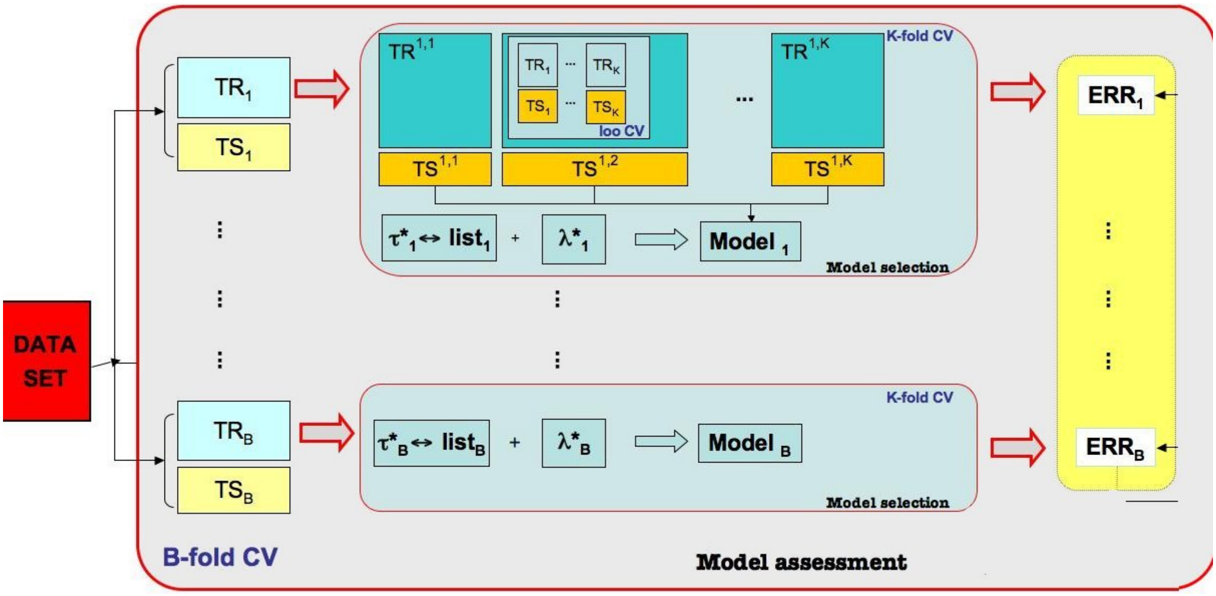


Figure 4.5: The structure of a bias-selection aware framework for variable selection when dealing with small data sets.

The protocol is based on two main steps: in the inner loop we select the relevant features and build the predictive model; in the outer loop we then assess its statistical robustness and significance whithin a complete validation framework. In details, given a dataset, we first split it in B subsplits, these subsplits will be used to assess the model. As shown in Figure 4.5, for each data set we evaluate the optimal regularization parameters τ_b^* and λ_b^* with $b = 1 \dots B$, by running the cross-validation loop described for Stage I for the intermediate data sample size using only the data set TR_b. This produces B models and B lists of relevant genes, and we can then estimate, in a honest way, the average error of our models over the (blind) test sets TS_b.

The gain in robustness provided by the double loop of cross-validation has an important drawback due to the non uniqueness of the estimator and the consequent lack of direct identifiability of a unique list of selected genes. In fact, in each split of the outer cross-validation loop an estimator is extracted, each depending on possibly different genes. However, although these gene lists can vary both in size and form, we can extract a unique list by selecting the most stable genes from the union of the B lists, i.e. the genes appearing with non null-weight at list a fixed number of times over the B splits. Furthermore, in this way, we can exploit an apparent weak point of the proposed protocol to assign to each gene a stability measure which allows us to extract rank lists of relevant genes to be handed to genetists and biolgists for further validation.

4.5 Algorithmic aspects

In the previous sections we referred to the solutions of ℓ^1, ℓ^2, and ℓ^1-ℓ^2 regularization in an abstract way, without providing any practical algorithm for computing them. In this section we described the algorithm that has been used to compute the solution of the three optimization principles, both in the toy problems described in this chapter and in the real data experiments which will be the main topic of the next chapter.

Concerning ℓ^2 regularization with linear model, it is a well-established results in statistics and inverse problems that the solution of (4.2) admits a closed form, given by

$$\beta = (\Psi^T\Psi + \lambda)^{-1}\Psi^T Y = \Psi^T(\Psi\Psi^T + \lambda)^{-1}Y.$$

In order to minimize (4.4), we use an algorithm which generalizes the following Landweber or gradient-descent iterative procedure [44], known to converge to a minimizer of the unpenalized least-squares objective function $\Phi_{\mu,\tau}(\beta) = \frac{1}{n}\left\|y - \Psi\beta\right\|_2^2$:

$$\beta^{p+1} = \beta^p + \frac{1}{C}[\Psi^T y - \Psi^T\Psi\beta^p]; \quad p = 0, 1, \ldots. \tag{4.5}$$

where the constant $2C$ is a strict upper bound for the spectral norm of the matrix $\Psi^T\Psi$: $\left\|\Psi^T\Psi\right\| < 2C$.

Inspired by the iterative thresholding algorithm proposed by [29] for pure ℓ^1 regularization, we propose a double modification of Landweber algorithm which provably converges to the minimizer of (4.4). The first modification amounts to applying a *soft-thresholding* operator $\mathbf{S}_{n\tau/C}$ at each iteration. The soft-thresholding operator $\mathbf{S}_\alpha$ acts on a vector component-wise as follows

$$[\mathbf{S}_\alpha(\beta)]_j = \begin{cases} (|\beta_j| - \alpha/2)\ \mathrm{sign}(\beta_j) & \text{if} \quad |\beta_j| \geq \alpha/2 \\ 0 & \text{if} \quad |\beta_j| < \alpha/2. \end{cases} \tag{4.6}$$

This operation enforces the sparsity of the regression coefficients in the sense that all coefficients below the threshold $\alpha/2$ are set to zero. The second modification is a simple multiplication which leads to the following damped iterative thresholding algorithm:

Algorithm 1 Damped Thresholded Landweber Algorithm

```
set $\beta^0 = 0$
for $p = 1, 2, \ldots,$ MAX_ITER do
    $\beta^{p+1} = \frac{1}{1+\frac{n\mu}{C}}\mathbf{S}_{\frac{n\tau}{C}}(\beta^p + \frac{1}{C}[\Psi^T y - \Psi^T\Psi\beta^p])$
end for
return $\beta^{\text{MAX\_ITER}}$.
```

We recover the cases of ridge regression and damped Landweber iteration for $\tau = 0$, whereas pure ℓ^1 regularization and the iterative thresholding scheme considered in [29] correspond to the special case $\mu = 0$. For $\tau = \mu = 0$, we get the original Landweber iteration (4.5). The convergence of Algorithm 1 – for $\mu > 0$ and any initial value β^0 – to the minimizer of (4.4) is a straightforward consequence of Banach's fixed point theorem for contractive mappings (see [30] for an extensive discussion of the properties of this algorithm in a broader setting).

We now need to define an efficient and easy-to-implement stopping rule for the iterative scheme. Clearly a pre-assigned maximum number of iterations is not a feasible choice. We thus tried to use a fixed tolerance δ, letting the iteration stop if $|\beta_j^{p+1} - \beta_j^p|_2 \leq \delta|\beta_j^p|_2$, for all j. However, after extensive experimentation on toy examples and real data, we empirically observed that a tolerance depending on the number of iterations is preferable. As shown in Figure 4.6, if the algorithm stops when the relative change of each coefficient β_k^p is smaller than a tolerance $\delta = 0.1/p$, with p the number of performed iterations, the support of the selected features is stabilized.

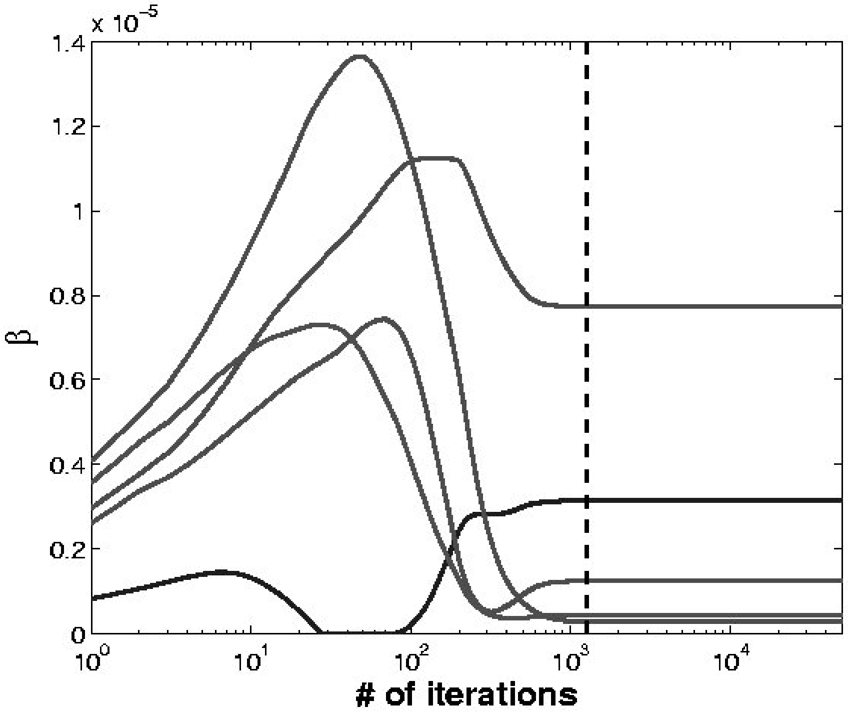


Figure 4.6: Coefficient path of the relevant genes for a real data set; the dashed line corresponds to the stopping rule.

The complexity of the iterative scheme described above is essentially governed by the cost of the Landweber iteration (possibly blockwise) times the number of iterations needed to reach convergence. As usual we assess the complexity of one Landweber iteration by estimating the required number of multiplications. This is the cost of the computation of $\Psi^T y - \Psi^T \Psi \beta^p$ which, alternatively, can be computed as $\Psi^T(y - \Psi\beta^p)$. In the first instance, we need to compute once for all and to store the vector $\Psi^T y$ (dn multiplications) and the $d \times d$ matrix $\Psi^T\Psi$ ($d^2 n$ multiplications). Then one iteration requires to multiply by this matrix the previous iterate of the coefficient vector, i.e. d^2 multiplications. When d becomes very large, the storage of the huge square matrix $\Psi^T\Psi$ may become problematic. Using the alternative implementation instead, we only need to store the rectangular $n \times d$ matrix Ψ and to compute $\Psi\beta$ (nd multiplications) and then to apply the transposed matrix to $y - \Psi\beta$, which again costs nd multiplications. This leads to a total cost of $2nd$ multiplications per iteration, which is more advantageous than the other scheme as soon as $d > 2n$. Notice also that only the n-dimensional vector y needs to be stored instead of the d-dimensional vector $\Psi^T y$. One should also add at each iteration, the cost for performing the componentwise thresholding as well as the multiplication by the damping factor. This cost however is not the main issue and can be neglected in first approximation. Also the normalization of the matrix (using for instance the power method) is done only once and is not significant as concerns our estimations of complexity.

The complexity of the single iteration is thus not very high, however, the overall complexity is governed by the number of iterations, which can be very large for small values of μ. Indeed small values of the ℓ^2 parameter are needed to perform first step for the construction of each classifier $\beta(t, l, i)$ in Stage I, necessary to compute the solution of Algorithm 1. Consequently, the procedure for determining the optimal values of τ and λ in Stage I, procedure which must be repeated $k \times |\mathcal{T}| \times |\mathcal{L}|$ times with $\mu = 10^{-6}$, is quite slow. Alternatively, we explored a different approach which exploits the almost perfect nesting properties of the one parameter family of solutions: for each value of τ and λ, the damped thresholding algorithm with the stopping rule described above is recursively run for 10 decreasing values of μ the i-th algorithm being restricted to the variables selected in previous $(i-1)$-th scheme (see Algorithm 2).

Algorithm 2 Accelerated ℓ^1-ℓ^2 regularization Algorithm

```
Given $\Psi, Y$
set $\mu_1 = 10^{-3}, ..., \mu_{10} = 10^{-6}$
set $\tilde{\beta}^{(0)} = \Psi^T(\Psi\Psi^T + \mu_1)^{-1}Y$
for $i = 1, \dots, 10$ do
    set  $C > 2\left\|\Psi^T\Psi\right\|$
         $\beta^0 = \tilde{\beta}^{(i-1)}$
         $p = 1$
    while $|\beta_j^{p+1} - \beta_j^p|_2 > \delta|\beta_j^p|_2$ for at least one $j$ do
        $$\beta^p = \frac{1}{1+\frac{n\mu}{C}}\mathbf{S}_{\frac{n\tau}{C}}(\beta^{(p-1)} + \frac{1}{C}[\Psi^T Y - \Psi^T\Psi\beta^{(p-1)}]),$$
        $p = p + 1$
    end while
    restrict $\Psi$ to the colums corresponding to the non null entries of $\beta^p$.
    set $\tilde{\beta}^{(i)} = \beta^p$ restricted to its non null entries.
end for

return $\tilde{\beta}^{(10)}$.
```

Although we do not have any theoretical argument showing the equivalence of the two methods, the experiments we present in the next Chapter indicate that the features selected through this alternative approach are almost always the same as those obtained with the procedure described in the original scheme, but with a reduction in computing time by a factor of about 100. We cannot predict in advance the overall time required to compute the complete set of 10 solutions $\beta^{(1)}, \dots, \beta^{(10)}$, but we report the times needed to run the accelerated ℓ^1-ℓ^2-regularization on a

number of samples	25	23	28	38	31
computing time (s)	304	38	240	136	57

Table 4.2: Computing times required to run the accelerated ℓ^1-ℓ^2-regularization described in Algorithm 2 on a set of typical microarray experiments.

set of typical microarray experiments with 54000 variables and number of samples about 30. Note that the computing times are not directly proportional to the number of samples, because they also strongly depend on the stability of the solution.

4.6 Toy experiment

In order to test our approach in a controlled setting, we applied the two stage ℓ^1-ℓ^2-regularization on a toy example generated according to scenario (d) in [122], where we know in advance which are the relevant or correlated features. The proposed toy problem is close to real gene expression data conditions, in that it encompasses both dependence on more than one variable, and intra-variables correlation, though in a lower dimensional setting. In details, a set of $n = 100$ toy-patients are drawn from $\mathbb{R}^d$ with $d = 40$ in the following way:

$$x_i = Z_1 + \epsilon_i^x, \quad Z_1 \sim N(0,1), \quad \text{for } i = 1,\dots,5,$$
$$x_i = Z_2 + \epsilon_i^x, \quad Z_2 \sim N(0,1), \quad \text{for } i = 6,\dots,10,$$
$$x_i = Z_3 + \epsilon_i^x, \quad Z_3 \sim N(0,1), \quad \text{for } i = 11,\dots,15,$$
$$x_i \sim N(0,1) \quad \text{for } i = 16,\dots,40,$$
$$y = x\beta + 15\epsilon, \quad \beta = (\underbrace{3,\dots,3}_{15},\underbrace{0,\dots,0}_{25})$$

where $\epsilon_i^x \sim N(0, 0.01)$ for $i = 1,\dots,15$, and $\epsilon \sim N(0,1)$. The above model consists of three equally relevant groups of variables ($G_1 = \{1,\dots,5\}, G_2 = \{6,\dots,10\}, G_3 = \{11,\dots,15\}$), where each group comprises five members, and a group of 25 pure noise features ($G_4 = \{16,\dots,40\}$). Since the variables in each of the first three groups are highly correlated and thus redundant, we expect that ℓ^1-reg will select just one variable per relevant group, and will discard features 16 through 40. We also expect that, by increasing the ℓ^2-reg parameter, the algorithm will selects also the other 12 features from the relevant groups.

The ℓ^1-reg is run over the first 50 samples for different values of τ and λ. The optimal ℓ^1 and ℓ^2 parameters, τ^* and λ^*, are chosen as the ones minimizing the error over the remainder 50 samples. ℓ^1-ℓ^2-reg is then run with $\mu = 1000 \cdot \tau^*$ in order to extract larger sets of features. Such a procedure is then performed over 50 independent data sets. The histogram of the selected models (features corresponding to non null coefficients) is drawn for $\mu = 0$, see Figure 4.7. Indeed the probability of selecting one feature from each of the first three groups is very high. Moreover, when the algorithm misses the true sparsest model, most times it selects either a redundant variable from G_1, G_2 or G_3 (bar "$[4,0]$"), or a noisy variable from G_4 (bar "$[3,1]$"), or both of them (bar "$[4,1]$"). The probability of selecting a model different from the ones above is smaller than 0.1. For $\mu = 1000 \cdot \tau^*$, we report the histogram of the number of selected features, Figure 4.8 (left) and of the ratio between the number of selected variables belonging to either G_1, G_2 or G_3, and the overall number of selected features, Figure 4.8 (right). Clearly

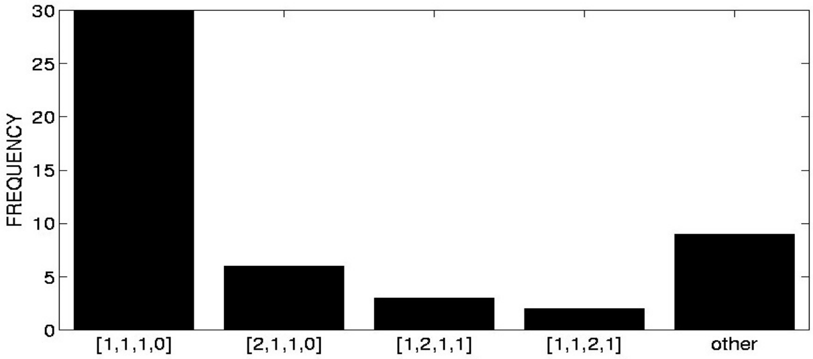


Figure 4.7: Histogram of selected models in toy problem for $\mu = 0$; $[1,1,1,0] = \{x_i, x_j, x_k\}, i \in G_1, j \in G_2, k \in G_3$, $[2,1,1,0] = \{x_{i_1}, x_{i_2}, x_j, x_k\}, i_1, i_2 \in G_1, j \in G_2, k \in G_3$, $[1,2,1,0] = \{x_i, x_{j_1}, x_{j_2}, x_k\}, i \in G_1, j_1, j_2 \in G_2, k \in G_3$, $[1,1,2,0] = \{x_i, x_j, x_{k_1}, x_{k_2}\}, i \in G_1, j \in G_2, k_1, k_2 \in G_3$, "other" = any other group of genes different from the above.

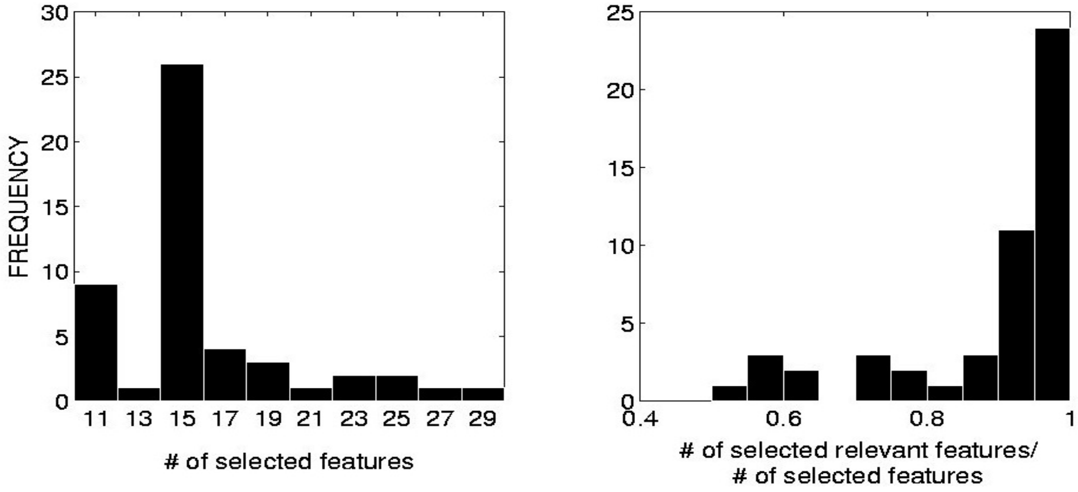


Figure 4.8: Histograms of the models selected for $\mu = 1000 \cdot \tau^*$; (left) overall number of selected features; (right) ratio between the number of selected variables belonging to either G_1, G_2 or G_3, and the overall number of selected features

the number of selected variables is peaked around 15, and from Figure 4.8 (right) we can see that most times such features truly relevant.

Chapter 5

Sparse regularization and Fenchel duality

In this chapter we propose a general framework to characterize and solve the optimization problems underlying a large class of sparsity based regularization algorithms. More precisely, we study the minimization of learning functionals that are sum of a differentiable data term and a convex non differentiable penalty. Non convex penalties have recently become popular since they allow to enforce some kind of sparsity in the solution. Leveraging on the theory of Fenchel duality and subdifferential calculus, we derive optimality conditions for the regularized solution and propose an extremely simple, yet general iterative projection algorithm whose convergence to the optimal solution can be proven. The power of the general framework is illustrated, considering several examples of regularization schemes including multi-task and multi-kernel learning, among others.

In Section 5.2, we begin by setting the notation and recalling some basic mathematical properties necessary to introduce the iterative algorithm and state its main properties.

In order make it easier for the reader to evaluate the contributions of our work, we state all the mathematical and algorithmic results first, and postpone the proofs to Section 5.3.

In Section 5.4, in order to show the potentially wide contribution of our work to practitioners of machine learning, we detail possible applications of the general algorithm to different interesting learning contexts.

5.1 Sparse Regularization

In learning from examples one tries to infer some quantity of interest, given a training set which is randomly sampled and corrupted by noise. In Chapter 3 and 4 we have seen that learning schemes which are simply tailored to minimize a data fit objective term, such as empirical risk minimization (ERM), typically lead to unstable solutions that do not generalize to new examples. Such an instability is often due to the ill-posedness of the optimization problem under study. An effective way to restore stability and find meaningful solutions is, therefore, to resort to regularization techniques. This class of methods typically involves the minimization of an objective function which is the sum of two terms. The first one is a data fit term, whereas the second is a penalty that favors "simple" models. Approaches based on Tikhonov regularization, including Support Vector Machines or Regularized Least Squares, are probably the most popular examples in this class of methods and are based on convex differentiable penalties.

In Chapter 4 we introduced a set of regularization methods, such as the *Lasso* [106] and variants like elastic net or *group lasso*, which recently received considerable attention because of their property to provide *sparse* solutions. The key towards sparsity properties is considering convex non differentiable penalties. More generally this kind of penalties have been used to deal with complex models for multi-task and multi-kernel learning. In this chapter we extend the concept of *sparsity based regularization* algorithms to the general class of methods using convex non differentiable penalties, and we study the problem of computing the regularized solution. The presence of a non differentiable penalty makes the solution of the minimization problem non trivial and recently there has been a considerable amount of work devoted to this problem, especially in the context of ℓ_1 regularization.

The presence of a non differentiable penalty makes the solution of the function minimization non trivial and the goal of this chapter is to provide a general framework to solve such a problem effectively. To this aim we assume that the penalty term is convex and exploit such an assumption to make use of tools from convex optimization and convex analysis. Then we can derive optimality conditions for the regularized solution of a large class of methods and propose an iterative algorithm for which we can prove convergence to such a regularized solution. As observed in (see [91]), Fenchel duality plays an important role in greatly simplifying the solution of minimization of regularized functional. The algorithm proposed in Section (5.2) only assumes that the functional to be minimized is strictly convex and differentiable (a related approach can be found also in [48] for a specific form of the functional). For specific minimization problems, in particular ℓ_1 regularization, many different techniques have been proposed to solve (5.1). Important examples are quadratic programming [27], LARS [38] and Bregman divergence [116]. Our approach is a suitable alternative to these techniques, moreover it might be of help for solving many different regularization problems involving non differentiable terms, since the theorems underpinning our framework hold under very broad assumptions.

The contribution of this chapter is, therefore, to show that, under fairly mild assumption on the penalty (it must be one-homogeous), sparsity based algorithms can be studied within a unifying framework that allows for the development of a simple iterative procedure to compute the regularized solution. Besides being very easy to implement, the proposed scheme is shown to converge to the optimal solution. Using Fenchel duality we decouple the contributions due to the data, and the penalty terms, briefly, at each iteration the gradient of the data term is projected on a set which is defined by the considered penalty. The iterative soft thresholding method recently proposed to solve the Lasso minimization can be recovered as a corollary of our results, that are flexible enough to account for a large class of problems. Besides this first result, we use this fact to derive new extremely simple optimization schemes for multi-task learning, multi-kernel learning, sparse principal component analysis and total variation regularization among others.

5.2 Iterative Projection Algorithm

After describing the general class of regularized learning algorithms under study, we proceed discussing the iterative procedure to compute the regularized solution and provide a detailed analysis. The latter consists in three main steps. First we show that the regularized solution satisfies a suitable fixed point equation involving a projection on a convex set, so that we can consider the iteration corresponding to the associated successive approximation scheme. Then we show how to compute the inner projection by generalizing previous results for total variation regularization. Finally we use the fixed point-theorem to prove convergence of the proposed procedure.

5.2.1 Setting

Given a a Hilbert space $\mathcal{H}$ and a fixed positive number τ, we consider the problem of computing:

$$f^* = \operatorname*{argmin}_{f\in\mathcal{H}} \Phi_{\mu,\tau}(f) = \operatorname*{argmin}_{f\in\mathcal{H}} F(f) + 2\tau\mathcal{J}(f), \tag{5.1}$$

where $F : \mathcal{H} \to \mathbb{R}$, $\mathcal{J} :\to \mathbb{R}\cup+\infty$ can be interpreted as the data and penalty terms, respectively. In the following, F is assumed to be differentiable and strictly convex, while $\mathcal{J}$ is required to be lower semicontinuous, convex, coercive and one-homogeneous,

$$\mathcal{J}(\lambda f) = \lambda\mathcal{J}(f),$$

for all $f \in \mathcal{H}$ and $\lambda \in \mathbb{R}^+$. Before presenting our results we give several examples for F and $\mathcal{J}$.
Loss term. In the context of supervised learning, given a training set $\{(x_i, y_i)\}_{i=1}^n \in \mathbb{R} \times Y$, $Y = [-M, M]$, the most common choice for the data term F is the empirical risk associated to some cost function $\ell : \mathbb{R} \times Y \to \mathbb{R}^+$, i.e.

$$F(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i),$$

with f in a suitable hypotheses space $\mathcal{H}$. Examples of convex and differentiable loss functions are the exponential loss $e^{-yf(x)}$, the logistic loss $\log(1+e^{-yf(x)})$, and the square loss $(y-f(x))^2$. In general, the corresponding empirical risk will be only convex, and strict convexity can be ensured under further assumptions on the data. An alternative way to enforce strict convexity is to add the strictly convex term $\mu \|f\|_{\mathcal{H}}^2$ for some small positive paramater μ. This can be seen as a preconditioning of the problem, and, if μ is small enough, one can see empirically that the solution does not change. Another important example for the data term is

$$F(f) = \|Af - y\|_{\mathcal{Y}}^2, \tag{5.2}$$

where $A : \mathcal{H} \to \mathcal{Y}$ is a bounded linear operator between Hilbert spaces $\mathcal{H}$, $\mathcal{Y}$, that might depend on the data, and $y \in \mathcal{Y}$ is a measurement function from which we aim at reconstructing f.

This latter choice is general enough to deal with eigen-problems underlying many unsupervised methods such as principal component analysis or spectral clustering.

Penalty term. The assumptions on the penalty – lower semicontinuity, coercivity, convexity and one-homogeneity – are satisfied by a general class of penalties that are sum of norms in distinct Hilbert spaces:

$$\mathcal{J}(f) = \sum_{k=1}^{p} ||J_k(f)||, \tag{5.3}$$

where, for all k, $J_k : \mathcal{H} \to \mathcal{H}_k$ is a bounded linear operator and $\|\cdot\|$ is the norm in H_k. This is the class of penalties we consider, the most common choice being the case $\mathcal{H}_k = \mathbb{R}^{m_k}$. For example, if the estimator is assumed to be described by a generalized linear model $f(x) = \sum_{j=1}^{p} \psi_j(x)\beta_j$, the ℓ_1 norm of the coefficients $J(\beta) = \sum_{j=1}^{p} |\beta_j|$ is a special case of the above penalty. If the coefficients are divided into "blocks", a penalty of the form (5.3), corresponding to the sum of the euclidean norm of each block, has been proposed in the so called group lasso and composite absolute penalties algorithms. Similar penalties have been used for multiple task learning (see the following) and sparse principal component analysis.

Another example is multiple kernel learning where the estimator is assumed to be $f = f_1 + \cdots + f_p$ and every f_j belongs to a specific RKHS $\mathcal{H}_j$ with kernel K_j and norm $\|\cdot\|_j$. In this case, the penalty term takes the form $\sum_{j=1}^{p} \|f_j\|_j$.

The above examples are only a few instances of methods satisfying the required assumptions, but one can see how loosely related learning schemes can be cast within a common general framework (see also Section 5.4). In the next section we show how the corresponding optimization problems can be solved using the same simple procedure.

5.2.2 Algorithm

In this section we describe the iterative procedure for computing the solution f^* of the convex minimization problem (5.1).

To this aim we recall some basic facts in convex analysis and introduce some definitions (see [43]). If $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ is a Hilbert space, the subdifferential at $f \in \mathcal{H}$ of a convex functional $Q : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ is denoted with $\partial Q(f)$ and is defined as the set

$$\partial Q(f) := \{h \in \mathcal{H} \, : \, Q(g) - Q(f) \geq \langle h, g - f \rangle_{\mathcal{H}} \quad \forall g \in \mathcal{H}\}.$$

If Q is not only convex but also differentiable, then the subdifferential reduces to a unique element which is precisely the gradient $\nabla Q(f)$ of Q at f. Given the above definition we let

$$K := \partial \mathcal{J}(0),$$

and denote with $\pi_{\lambda K} : \mathcal{H} \to \mathcal{H}$ the projection on $\lambda K \subset \mathcal{H}$, $\lambda \in \mathbb{R}^+$ (which is well defined since the subdifferential is always a convex and closed set, and it is nonempty because $J(0) = 0$).

Given the above definitions the optimization scheme we derive is given by Algorithm 3. The parameter σ can be seen as a step-size, whose choice is crucial to ensure convergence and is discussed in the following. As we mentioned before, our approach decouples the contributions of the two functionals J and F. At each iteration of the algorithm, the projection $\pi_{\lambda K}$– which is

Algorithm 3 General Algorithm

```
initialize $\sigma, \tau > 0$
set $f^0 = 0$
while stopping criterion do
  $p = p + 1$
  $$f^p = \left(I - \pi_{\frac{\tau}{\sigma}K}\right)\left(f^{p-1} - \frac{1}{2\sigma}\nabla F(f^{p-1})\right) \quad (5.4)$$
end while
```

entirely characterized by $\mathcal{J}$ – is applied to a term that depends only on F. Fenchel duality ([43]) is the key tool that, combined with one-homogeneity, allows us to characterize the contribution of $\mathcal{J}$. We note that this line of reasoning is developed in a systematic way in the so called *forward-backward splitting* approach REF.

In the following we state and prove the key results toward deriving Algorithm 3.

5.2.3 Fixed Point Equation

We start showing that the optimal solution of problem (5.1) is the unique fixed point of a family of functionals parameterized by the step size σ.

Theorem 1. *Given $\tau > 0$, $F : \mathcal{H} \to \mathbb{R}$ strictly convex and differentiable and $\mathcal{J} : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ lower semicontinuous, coercive, convex and one-homogeneous, the minimizer f^* of $\Phi_{\mu,\tau}$ is the unique fixed point of the map $\mathcal{T}_\sigma : \mathcal{H} \to \mathcal{H}$ defined by*

$$\mathcal{T}_\sigma(f) = \left(I - \pi_{\frac{\tau}{\sigma}K}\right)\left(f - \frac{1}{2\sigma}\nabla F(f)\right). \tag{5.5}$$

We postpone the proof to Section 5.3, but it is worth remarking that strict convexity of F is assumed only to ensure uniqueness of the minimizer of $\Phi_{\mu,\tau}$, and that the fixed point equation is indeed satisfied by each minimizer of $\Phi_{\mu,\tau}$ in the case F is merely convex.

We note that, Algorithm 3 is simply the successive approximation scheme associated to the above fixed point equation. If the map $\mathcal{T}_\sigma$ is a contraction convergence of the iteration is ensured by Banach fixed point theorem and convergence rates can be easily obtained. Recall that we say that a map $\mathcal{T}_\sigma$ is a contraction if

$$|\mathcal{T}_\sigma(f) - \mathcal{T}_\sigma(g)| \leq L_\sigma \|f - g\|, \qquad \forall f, g \in \mathcal{H}$$

and $L_\sigma < 1$. In fact, in our setting $\mathcal{T}_\sigma$ depend on σ that we can choose so that $L_\sigma < 1$. In this case the following inequality relates the solution f^p at iteration step p and the solution f^* of the minimization problem,

$$\|f^* - f^p\| \leq \frac{L_\sigma^p}{1 - L_\sigma}\|f^1 - f^0\|.$$

The constant L_σ depends only on the data fit term as can be seen by the following result.

Proposition 1. *Assume the penalty term to satisfy the assumptions in Theorem 1 and F to be twice differentiable with continuous second derivative $\nabla^2 F : \mathcal{H} \to \mathcal{L}(\mathcal{H}, \mathcal{H})$.*

Moreover let $a(f) \geq b(f)$ denote the largest and smallest eigenvalues of $\nabla^2 F(f)$ and assume that there exist $a > b > 0$ such that $a \geq a(f) \geq b(f) \geq b$ for all $f \in \mathcal{H}$. Then the map $\mathcal{T}_\sigma$ is a contraction if we choose σ such that

$$\max\left\{\left|1 - \frac{a}{2\sigma}\right|, \left|1 - \frac{b}{2\sigma}\right|\right\} < 1. \tag{5.6}$$

The optimal a priori choice for the step size is given by

$$\sigma = \frac{a+b}{4}$$

and in this case $L_\sigma = \frac{a-b}{a+b}$.

Again, we postpone the proof to Section 5.3 and explicitly compute L_σ in several cases in Section 5.2.5. In the above theorem $\nabla^2 F : \mathcal{H} \to \mathcal{L}(\mathcal{H}, \mathcal{H})$ denotes the second derivative of F. To write it in this form, with an abuse of notation, we implicitly identified the linear operator $\nabla F : \mathcal{H} \to \mathcal{L}(\mathcal{H}, \mathbb{R})$ with an element of $\nabla F \in \mathcal{H}$, and then we computed the second derivative (see [71] for more details).

The above result shows that in general, for a strictly convex F, if the condition on the smallest eigenvalue of the second derivative is not uniformly bounded from below by a strictly positive constant, it might not be possible to choose σ so that $L_\sigma < 1$. The next corollary shows that this can always be done if F is perturbed by adding the term $\mu \|\cdot\|_{\mathcal{H}}^2$, with $\mu > 0$.

Corollary 1. *Assume the penalty term to satisfy the assumptions in Theorem 1 and F to be convex and twice differentiable with continuous second derivative $\nabla^2 F$. Moreover let $a(f) \geq b(f) \geq 0$ denote the largest and smallest eigenvalues of $\nabla^2 F(f)$ and suppose that $a(f) \leq a$. Consider the perturbed function $F_\mu = F + \mu \|\cdot\|_{\mathcal{H}}^2$, with $\mu > 0$ and set $b = \inf_{f \in \mathcal{H}} b(f)$. Then the map $\mathcal{T}_\sigma$ obtained in correspondence of F_μ is a contraction if we choose σ such that*

$$\max\left\{\left|1 - \frac{\mu}{\sigma} - \frac{a}{2\sigma}\right|, \left|1 - \frac{\mu}{\sigma} - \frac{b}{2\sigma}\right|\right\} < 1. \tag{5.7}$$

The optimal a priori choice for the step size is given by

$$\sigma = \frac{a+b}{4} + \mu,$$

and in this case $L_\sigma = \frac{a-b}{a+b+4\mu}$.

The above corollary highlights the role of the term $\mu \|\cdot\|_{\mathcal{H}}^2$ as a natural preconditioning of the algorithm. One can also argue that, if μ is chosen small enough, the solution is expected not to change and in fact converges to a precise minimizer of $F + J$. Indeed, the quadratic term performs a further regularization that allows to select, as μ approaches 0 , the minimizer of $F + J$ having minimal norm (see for instance [36]).

We add two remarks. First, we note that, in certain cases, it is still possible to prove convergence of the above iteration also for $\mu = 0$, though the analysis is considerably more complicated- see [29, 47] Second, the a-priori step-size choice discussed above is only one of the possible step-size choice strategy and allows for a simple convergence analysis. More aggressive and iteration dependent step-choices are likely to considerably speed-up convergence. In the next section we discuss how to compute the projection π_K.

5.2.4 Computing the Projection

We discuss how to compute the projection π_K when $\mathcal{J}$ is of the form

$$\mathcal{J}(f) = \sum_{k=1}^{p} ||J_k(f)||_k, \tag{5.8}$$

where, for all $k = 1, \ldots, p$, $\mathcal{G}_k$ is a Hilbert space with norm $\|\cdot\|_k$ and $J_k : \mathcal{H} \to \mathcal{G}_k$ is a bounded linear operator.

In the following proposition we characterize the set $\partial\mathcal{J}(0)$ and give a useful representation of the projection on this set.

Proposition 2. *Let $\mathcal{J}(f)$ as in (5.8) and*

- *$\mathcal{G} = \prod_{k=1}^{p} \mathcal{G}_k$, so that $v = (v_1, \ldots, v_p) \in \mathcal{G}$ with $v_k \in \mathcal{G}_k$ and $\|v\| = \sum \|v_k\|_k$;*
- *$J : \mathcal{H} \to \mathcal{G}$ such that $J(f) = (J_1(f), \ldots, J_p(f))$.*

Then

$$\partial\mathcal{J}(0) = \{J^T v : v \in \mathcal{G}, \|v_k\|_k \leq 1 \ \forall k\},$$

where $J^T : \mathcal{G} \to \mathcal{H}$ is the adjoint of J, and can be written as $J^T v = \sum_{k=1}^{p} J_k^T v_k$.
Moreover the projection of an element $g \in \mathcal{H}$ on the set $\lambda K := \lambda\partial\mathcal{J}(0)$ is given by $\lambda J^T \bar{v}$, where

$$\bar{v} \in \operatorname*{argmin}_{v \in \mathcal{G},\ \|v_k\|_k \leq 1} \left\|\lambda J^T v - g\right\|_{\mathcal{H}}^2. \tag{5.9}$$

We refer to Section 5.3 for the proof of the above result. Note that even though the definition of $\bar{v}$ may not be unique, if J has non trivial null space, the definition of the projection $\pi_{\lambda K}(g)$ is always uniquely defined.
As we will discuss in the following, in several specific cases the nonlinear projection π_K can be expressed analytically in a closed form. Nonetheless, in general its computation is not straightforward. An efficient solution to an analogue problem has been recently proposed in the context of total variation image denoising [26]. We generalize this latter approach to derive an iterative scheme for computing the solution of problem (5.9) induced by penalties $\mathcal{J}$ of the form (5.8). Towards this end, we note that the Karush-Kuhn-Tucker conditions associated to (5.9) ensure the existence of a set of Lagrange multipliers α_k, such that for all k

$$J_k(\lambda J^T v - g) + \alpha_k v_k = 0,$$

with either $\|v_k\|_k = 1$ and $\alpha_k > 0$, or $\|v_k\|_k < 1$ and $\alpha_k = 0$. In both cases v_k satisfies

$$J_k(\lambda J^T v - g) + \left\|J_k(\lambda J^T v - g)\right\|_k v_k = 0 \quad \forall k. \tag{5.10}$$

The above equation leads to a fixed point equation which solution can be computed by means of the iteration given in the theorem below.

Theorem 2. *Given $\mathcal{J}$ as in Proposition 2, let $\kappa = (\left\|JJ^T\right\|)$ and $\eta = (\left\|JJ^T\right\|)^{-1}$, $v^0 = 0$ and for any $q \geq 0$, set*

$$v_k^{q+1} = \frac{v_k^q - \eta J_k\left(J^T v^q - g/\lambda\right)}{1 + \eta \left\|J_k\left(J^T v^q - g/\lambda\right)\right\|_k}. \tag{5.11}$$

Then $\left\|\lambda J^T v^q - \pi_{\lambda K}(g)\right\|_{\mathcal{H}}$ converges to 0 as $q \to \infty$.

Again, the proof is given in Section 5.3 and the explicit form of the projection for several different examples is discussed in Section 5.4. We remark that the convergence in the above result refers to the projection rather than to the possibly not unique function $\bar{v}$. Before dealing with examples, we discuss convergence and step size choice for Algorithm 3.

5.2.5 Computing the Step Size

We discuss the step size choice in two specific setting of interests. First, we consider supervised learning problems where, given a training set $\{(x_i, y_i)\}_{i=1}^n$, with $x \in X \subset \mathbb{R}^d$ and $y \in Y = [-M, M]$, we have to find an unknown functional relation $f : X \to Y$. We consider loss functions $\ell : \mathbb{R} \times Y \to \mathbb{R}^+$ that are convex and twice differentiable in the first argument. Moreover we consider functions belonging to a RKHS [6]. In particular we make use of the following well known facts. A function f in a RKHS $\mathcal{H}$ with kernel K, can be seen as a hyperplane $f(x) = \langle \Phi(x), \beta \rangle_{\mathcal{F}}$, where $(\mathcal{F}, \langle \cdot, \cdot \rangle_{\mathcal{F}})$ is a Hilbert space- *the feature space*, $\beta \in \mathcal{F}$ and $\Phi : X \to \mathcal{F}$ is called feature map and satisfies

$$\left\langle \Phi(x), \Phi(x') \right\rangle_{\mathcal{F}} = K(x, x').$$

In particular we make use of the following properties, $\forall f \in \mathcal{H}$, $\|f\|_{\mathcal{H}} \leq \|\beta\|_{\mathcal{F}}$ and

$$\sup_{x \in X} |f(x)| \leq \kappa \|\beta\|_{\mathcal{F}},$$

where for the latter inequality to hold true, we need to assume that $\sup_{x \in X} \|\Phi(x)\| \leq \kappa$, (the kernel is bounded). In the following we consider in particular two examples of feature maps. The first is given by the reproducing kernel K setting $\Phi(x) = K(x, \cdot)$ so that $(\mathcal{F}, \langle \cdot, \cdot \rangle_{\mathcal{F}})$ is simply $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ and $f = \beta$, implying $\|f\|_{\mathcal{H}} = \|\beta\|_{\mathcal{F}}$. The second example corresponds to considering a finite set of functions (a dictionary) $(\psi_j)_{j=1}^p$ and setting $\Phi(x) = (\psi_1(x), \ldots, \psi_p(x))$ so that $\mathcal{F}$ can be indentifed with $\mathbb{R}^p$ with the corresponding inner product. In this case $\|f\|_{\mathcal{H}} \leq \|\beta\|_{\mathcal{F}}$ where the equality holds if the dictionary is an orthonormal basis.

Given the above premises, the specific data terms F we consider can be written as

$$F(\beta) = \sum_{i=1}^n \ell(\langle \Phi(x_i), \beta \rangle_{\mathcal{F}}, y_i) + \mu \|\beta\|_{\mathcal{F}}^2. \tag{5.12}$$

where $\mu \geq 0$. [1]

The following result studies the property of the map $\mathcal{T}_\sigma$ induced by the above functional, and in particular provides the optimal choice for the step-size σ using the result in Proposition 1. We show that the optimal σ is entirely determined by the covariance operator, which is defined as

$$\begin{array}{rccl} \mathrm{Cov}: & \mathcal{F} & \to & \mathcal{F} \\ & \beta & \mapsto & \sum_{i=1}^n \langle \Phi(x_i), \beta \rangle \, \Phi(x_i). \end{array}$$

It is well-known that Cov is symmetric and positive, so that denoting by a and b the largest and the smallest eigenvalues, it follows $a \geq b \geq 0$.

Proposition 3. *Assume the penalty term to satisfy the assumptions in Theorem 1 and F to be given by (5.12). Moreover let a and b denote the largest and smallest eigenvalues of* Cov *and $0 \leq L_{min} \leq \ell''(w,y) \leq L_{max}$, $\forall w \in \mathbb{R}, y \in Y$, where ℓ'' denotes the second derivative of ℓ with respect to w. Then the map $\mathcal{T}_\sigma$ is a contraction of constant L_σ if we choose σ such that*

$$\max\left\{ |1 - \frac{\mu}{\sigma} - \frac{L_{max} a}{2\sigma}|, |1 - \frac{\mu}{\sigma} - \frac{L_{min} b}{2\sigma}| \right\} < 1. \tag{5.14}$$

The optimal a priori choice for the step size is given by

$$\sigma = \frac{aL_{max} + bL_{min}}{4} + \mu,$$

and in this case $L_\sigma = \frac{aL_{max} - bL_{min}}{aL_{max} + bL_{min} + 4\mu}$.

We give the proof of the above result in Section 5.3. Note again that if we let μ be equal to zero, then equation (5.14) may be never satisfied when either $L_{\min}$ or b are zero.

We add examples of for specific loss functions.

Example 1 (Square Loss)**.** *Consider the square loss $\ell(w,y) = (w-y)^2$. Then $\ell''(w,y) = L_{min} = L_{max} = 2$ $\forall w \in \mathbb{R}, y \in Y$ and the optimal a priori choice for the step size is given by $\sigma = \frac{a+b+2\mu}{2}$.*

Example 2 (Exponential Loss)**.** *If we consider the exponential loss $\ell(w,y) = e^{-wy}$, then $\ell''(w,y) = y^2 e^{-wy}$. Since $Y = [-M, M]$ we can assume without loss of generality that $f(x) \in [-M, M]$ $\forall x$, so that $0 \leq \ell''(w,y) \leq M^2 e^{M^2}$. The optimal a priori choice for the step size is then given by $\sigma = \frac{aM^2 e^{M^2}}{4} + \mu$.*

Next we consider a data term of the form (5.2). More precisely, given two Hilbert spaces $\mathcal{H}, \mathcal{Y}$, and a bounded operator $A : \mathcal{H} \to \mathcal{Y}$, we consider

$$F = \|Af - y\|_{\mathcal{Y}}^2 + \mu \|f\|_{\mathcal{H}}^2 \tag{5.15}$$

which is strictly convex if $\mu > 0$ *or* A is injective. In particular, when $A = I$ the equation $f = \mathcal{T}_\sigma(f)$ admits an explicit solution f^*, which is unique even when $\mu = 0$. In fact, since $\frac{1}{2}\nabla F(f) = (1+\mu)f + y$, by setting $\sigma = 1 + \mu$, we obtain

$$f^* = \frac{y}{1+\mu} - \pi_{\frac{\tau}{1+\mu}K}\left(\frac{y}{1+\mu}\right) = \frac{1}{1+\mu}\left(y - \pi_{\tau K}(y)\right).$$

[1] Clearly if we choose $\Phi(x) = K(x, \cdot)$ we have

$$F(\beta) = F(f) = \sum_{i=1}^n \ell(f(x_i), y_i) + \mu \|f\|_{\mathcal{H}}^2 . \tag{5.13}$$

For a general operator A, the solution of $f = \mathcal{T}_\sigma(f)$ does not admit a closed form. However we can compute it using Algorithm 3, provided that the map $\mathcal{T}_\sigma$ is a contraction.

Proposition 4. *Assume the penalty term to satisfy the assumptions in Theorem 1 and F to be given by (5.15). Let a and b be the smaller and larger eigenvalues of $A^T A$, where A^T denotes the adjoint of A. Then the map $\mathcal{T}_\sigma$ is a contraction if we choose σ such that*

$$max\left\{\left|1 - \frac{a+\mu}{\sigma}\right|, \left|1 - \frac{b+\mu}{\sigma}\right|\right\} < 1.$$

The optimal a priori choice for the step size is given by $\sigma = \frac{a+b+2\mu}{2}$, and in this case $L_\sigma = \frac{a-b}{a+b+2\mu}$.

5.3 Proofs

In this section we collect the proofs of the results in the previous sections. We start by proving Theorem 1. The proof requires a few basic concepts from convex analysis [43]. In particular we recall that the Fenchel conjugate of a convex functional is defined as

$$\begin{array}{rcl} \mathcal{J}^* \; : \; \mathcal{H} & \rightarrow & \mathbb{R} \cup \{+\infty\} \\ g & \mapsto & \sup_{f \in \mathcal{H}} \langle f, g \rangle_{\mathcal{H}} - \mathcal{J}(f), \end{array}$$

and satisfies the well known Young-Fenchel equality:

$$g \in \partial \mathcal{J}(f) \iff f \in \partial \mathcal{J}^*(g). \tag{5.16}$$

The above equality is the key for the proof of Theorem 1 and leads to a dual formulation of the minimization problem (5.1). Another important fact is that the conjugate of a one-homogeneous functional $\mathcal{J}$ is the indicator function of the convex set $K = \partial \mathcal{J}(0)$ and this implies that the solution of the dual problem reduces to the projection onto K. In the proof of Proposition 2, we are also going to use some standard properties of the subdifferential, that can be found in [43], Chapter 1. For the convenience of the reader we recall them here, without stating all the needed assumptions that are sistematically satisfied in our setting.

P1) *Sum rule*: if F and J are convex, then $\partial(F + J)(f) = \partial F(f) + \partial J(f)$;

P2) *Chain rule*: let L be a linear operator and F a convex function, then

$$\partial(F \circ L)(f) = L^T(\partial F(L(f)))$$

P3) *Subdifferential of the norm in a Hilbert space H*:

$$(\partial \|\cdot\|)(0) = \{v \in H : \|v\| \leq 1\} := B(H, 1).$$

We can now give the proof of Theorem 1.

Theorem 1. Since $\Phi_{\mu,\tau}$ is lower semicontinuous, strictly convex and coercive, it admits a unique minimizer, which is characterized by the Euler equation

$$0 \in 2\tau\partial\mathcal{J}(f) + \nabla F(f).$$

Using (5.16) this is equivalent to

$$f \in \partial\mathcal{J}^*\left(-\frac{1}{2\tau}\nabla F(f)\right).$$

If we let $g = (f - \frac{1}{2}\nabla F(f))$, and add g/τ to both sides of the above relation, then we obtain

$$0 \in \frac{1}{\tau}(g-f) - \frac{g}{\tau} + \frac{1}{\tau}\partial\mathcal{J}^*\left(\frac{1}{\tau}(g-f)\right).$$

It follows that $w = \frac{1}{\tau}(g-f)$ is the minimizer of

$$\frac{1}{2}\left\|w' - \frac{g}{\tau}\right\|_{\mathcal{H}}^2 + \frac{1}{\tau}\mathcal{J}^*(w').$$

Since the penalty is one-homogeneus its Fenchel conjugate $\mathcal{J}^*$ is the indicator function of K, and we obtain

$$w = \underset{w' \in K}{\operatorname{argmin}} \left\|w' - \frac{g}{\tau}\right\|_{\mathcal{H}}^2 = \pi_K\left(\frac{g}{\tau}\right),$$

which immediately gives $f = g - \tau\pi_K\left(\frac{g}{\tau}\right) = g - \pi_{\tau K}(g)$. We conclude noting that we can multiply both F and τ by $\sigma > 0$, without modifying the minimizer of (5.1), which is therefore the unique fixed point of the mapping $\mathcal{T}_\sigma : \mathcal{H} \to \mathcal{H}$

$$\mathcal{T}_\sigma(f) = f - \frac{1}{2\sigma}\nabla F(f) - \pi_{\frac{\tau}{\sigma}K}\left(f - \frac{1}{2\sigma}\nabla F(f)\right),$$

and this ends the proof. □

Next, we prove convergence and step-size choice in the general case.

Proposition 1. We first observe that the contraction $\mathcal{T}_\sigma$ can be decomposed as $\mathcal{T}_\sigma = (I - \pi_{\frac{\tau}{\sigma}K}) \circ B_\sigma$, with $B_\sigma(f) := f - \frac{1}{2\sigma}\nabla F(f)$. Since $(I - \pi_{\frac{\tau}{\sigma}K})$ has unitary Lipshitz constant as an immediate consequence of the projection theorem, it is enough to prove that the inner mapping B_σ is a contraction. According to a corollary of the Mean Value Theorem (see Corollary 4.3 of [71] for the infinite dimensional version), every Fréchet differentiable mapping B such that $\sup_{f\in\mathcal{F}} \|B'(f)\| < 1$ is a contraction, therefore it is enough to prove that the norm of B'_σ is bounded by the unit. We have:

$$B'_\sigma(f) = I - \frac{1}{2\sigma}\nabla^2 F(f),$$

therefore

$$\|B'_\sigma\| \leq \max\left\{\left|1 - \frac{1}{2\sigma}a\right|, \left|1 - \frac{1}{2\sigma}b\right|\right\}.$$

Since $a \geq b > 0$ the r.h.s is strictly less than 1 and the first part of the thesis follows. The minimization of the function $\sigma \mapsto \max\{\left|1 - \frac{1}{2\sigma}a\right|, \left|1 - \frac{1}{2\sigma}b\right|\}$ gives the best a priori choice of σ, that is $\sigma = \frac{a+b}{4}$. □

Corollary 1. It is enough to note that $\nabla^2 F_\mu = \nabla^2 F + \mu I$, implying that the smallest eigenvalue of $F + \mu \left\|\cdot\right\|_{\mathcal{H}}^2$ is uniformly bounded from below by μ. The rest of the thesis easily follows applying Proposition 1 to F_μ. □

Next we consider the results allowing to compute the projection. First we prove Proposition 2.

Proposition 2. Using properties (P1) and (P2) stated at the beginning of the Section, and setting $K = \partial\mathcal{J}(0)$, we have

$$K = \sum_{k=1}^{p} \left(\partial \left(f \mapsto \left\|J_k f\right\|_k\right)\right)(0) = \sum_{k=1}^{p} J_k^T \left(\partial \left\|\cdot\right\|_k\right)(0)$$

where thanks to property (P3), $\left(\partial \left\|\cdot\right\|_k\right)(0) = \{v_k \in \mathcal{G}_k : \left\|v_k\right\|_k \leq 1\}$. Then we can identify the set K with

$$K = \{J^T v : \, v \in \mathcal{G}, \left\|v_k\right\|_k \leq 1 \;\forall k\}.$$

The projection on λK is then defined as $\pi_{\lambda K}(g) = \lambda J^T \bar{v}$, where $\bar{v}$ is given by (5.9). □

Then we prove Theorem 2.

Theorem 2. Equation (5.10) holds also if we multiply by $-\eta$ with $\eta > 0$ and add v_k to both sides, hence obtaining

$$-\eta \left(J_k(\lambda J^T v - g) + \left\|J_k(\lambda J^T v - g)\right\|_k v_k\right) + v_k = v_k,$$

so that v_k satisfies the fixed point equation

$$v_k = \frac{v_k - \eta J_k \left(J^T v - g/\lambda\right)}{1 + \eta \left\|J_k \left(J^T v - g/\lambda\right)\right\|_k}.$$

By induction it is easy to see that $\left\|v_k^q\right\|_k \leq 1$, for all k, q. We then introduce $h^q = (h_1^q, \ldots, h_p^q)$ and $\rho^q = (\rho_1^q, \ldots, \rho_p^q)$ with $h^q, \rho^q \in \mathcal{G}$ such that $h_k^q = J_k(J^T v^q - g/\lambda) \in \mathcal{G}_k$ and $\rho_k^q = \left\|h_k^q\right\| v_k^{q+1} \in \mathcal{G}_k$, so that $v_k^{q+1} = v_k^q - \eta(h_k^q + \rho_k^q)$.

$$\begin{aligned}
&\left\|J^T v^{q+1} - \tfrac{g}{\lambda}\right\|_{\mathcal{H}}^2 - \left\|J^T v^q - \tfrac{g}{\lambda}\right\|_{\mathcal{H}}^2 = \\
&\left\|J^T(v^q - \eta(h^q + \rho^q)) - \tfrac{g}{\lambda}\right\|_{\mathcal{H}}^2 - \left\|J^T v^q - \tfrac{g}{\lambda}\right\|_{\mathcal{H}}^2 = \\
&-2\eta \left\langle J^T(h^q + \rho^q), J^T v^q - \tfrac{g}{\lambda}\right\rangle_{\mathcal{H}} + \eta^2 \left\|J^T(h^q + \rho^q)\right\|_{\mathcal{H}}^2 = \\
&-2\eta \left\langle h^q + \rho^q, h^q\right\rangle + \eta^2 \left\|J^T(h^q + \rho^q)\right\|_{\mathcal{H}}^2 = \\
&-\eta \left\|h^q + \rho^q\right\|^2 - \eta \left\langle h^q + \rho^q, h^q - \rho^q\right\rangle + \eta^2 \left\|J^T(h^q + \rho^q)\right\|_{\mathcal{H}}^2 \leq \\
&-\eta \left[(1 - \eta\kappa) \left\|h^q + \rho^q\right\|^2 + (\left\|h^q\right\|^2 - \left\|\rho^q\right\|^2)\right] = \\
&-\eta \textstyle\sum_{k=1}^{p} \left[(1 - \eta\kappa) \left\|h_k^q + \rho_k^q\right\|_k^2 + (\left\|h_k^q\right\|_k^2 - \left\|\rho_k^q\right\|_k^2)\right].
\end{aligned}$$

The r.h.s in the above equation is a sum of p nonegative terms:

$$\underbrace{(1 - \eta\kappa) \left\|h_k^q + \rho_k^q\right\|_k^2}_{(1)} + \underbrace{(\left\|h_k^q\right\|_k^2 - \left\|\rho_k^q\right\|_k^2)}_{(2)}$$

In fact, (1) is clearly nonnegative for $\eta \leq \kappa$, whereas (2) ≥ 0 since $\left\|v_k^{q+1}\right\|_k \leq 1$ which implies $\|\rho_k\|_k \leq \|h_k\|_k$ We now examine the case where the $\left\|J^T v^{q+1} - \frac{g}{\lambda}\right\|_{\mathcal{H}}^2 - \left\|J^T v^q - \frac{g}{\lambda}\right\|_{\mathcal{H}}^2 = 0$. This requires both (1) and (2) to be null for all k. When $\eta < \kappa$, (1) $= 0$ only if $\left\|h_k^q + \rho_k^q\right\|_k = 0$ which implies both (2) $= 0$ and $v_k^{q+1} = v_k^q$. When $\eta = \kappa$, (1) is clearly null whereas (2) $= 0$ only if $\left\|h_k^q\right\|_k = \left\|\rho_k^q\right\|_k$ for all k which again implies $v_k^{q+1} = v_k^q$. Hence if $\eta \leq \left\|JJ^T\right\|^{-1}$, either $\left\|J^T v^q - g/\lambda\right\|_{\mathcal{H}}$ is decreasing or $v^{q+1} = v^q$.
Let $m = \lim_{n\to\infty} \left\|J^T v^q - g/\lambda\right\|$, and $\bar{v}$ be the limit of a converging subsequence (v^{q_l}) of (v^q). Clearly we have $m = \left\|J^T \bar{v} - g/\lambda\right\| = \left\|J^T \bar{v}' - g/\lambda\right\|$, where $\bar{v}'$ is the limit of (v^{q_l+1}). From the above calculations we see that since $\left\|J^T \bar{v}' - g/\lambda\right\| - \left\|J^T \bar{v} - g/\lambda\right\| = 0$, it must be $\bar{v}_k = \bar{v}'_k$ $\forall k$. Hence $\bar{v}$ satisfies the Euler equation (5.10) and therefore solves (5.9). Since the projection is unique, we deduce that all the sequence $\lambda J^T v^q$ converges to $\pi_{\lambda K}(g)$. □

Proposition 3. In order to apply Proposition 1, it is enough to show that the conditions on the eigenvalues of the second derivative of F are satisfied. Using the same notations as in Proposition 1, and relying on the chain rule (see [71]) we are able to explicitely compute the function B_σ for this specific choice of F, obtaining

$$B_\sigma(\beta) = (1 - \frac{\mu}{\sigma})\beta - \frac{1}{2\sigma}\sum_{i=1}^{n} l'(\langle\Phi(x_i), \beta\rangle, y_i)\Phi(x_i).$$

Reasoning as in the previous step, and again relying on the chain rule, we get

$$B'_\sigma(\beta)(\beta') = \left(1 - \frac{\mu}{\sigma}\right)\beta' - \frac{1}{2\sigma}A_\beta(\beta') \tag{5.17}$$

where $A_\beta(\beta') := \sum_{i=1}^{n} l''(\langle\Phi(x_i), \beta\rangle, y_i)\langle\Phi(x_i), \beta'\rangle\Phi(x_i)$. We note that A_β is a positive, self-adjoint linear operator thanks to the convexity of l, so that $a \geq b \geq 0$. In particular, using the fact that $L_{max}a$, $L_{min}b$ are respectively an upper and a lower bound of the eigenvalues of $\frac{1}{2\sigma}A_\beta$, we get the desired inequality and the optimal step choice. □

Finally we study the general least squares case. Although it can be viewed as a consequence of Proposition 1, we prefer to derive the desired inequality directly from the definition of $\mathcal{T}_\sigma$.

Proof of Proposition 4.

$$\begin{array}{rcl} \|\mathcal{T}_\sigma(f) - \mathcal{T}_\sigma(f')\| & = & ||(I - \pi_{\frac{\tau}{\sigma}K})(f - \frac{1}{2\sigma}\nabla F(f)) - (I - \pi_{\frac{\tau}{\sigma}K})(f' - \frac{1}{2\sigma}\nabla F(f'))|| \\ & \leq & ||f - \frac{1}{2\sigma}\nabla F(f) - f' + \frac{1}{2\sigma}\nabla F(f')|| \\ & \leq & ||I - \frac{1}{\sigma}(A^T A + \mu)||\,\|f - f'\| \\ & = & \max\left\{\left|1 - \frac{a+\mu}{\sigma}\right|, \left|1 - \frac{b+\mu}{\sigma}\right|\right\}\|f - f'\| \\ & =: & L_\sigma \|f - f'\|. \end{array}$$

The optimal a priori choice for the step size is given by the value of σ minimizing L_σ, that is

$$\sigma = \frac{a + b + 2\mu}{2},$$

and one can simply verify that $L_\sigma = \frac{a-b}{a+b+2\mu}$. □

5.4 Examples

In this section we derive important special cases of the general Algorithm 3, by specializing our analysis in a number of well known actual regularization schemes. First, we consider a group of algorithms which have become popular among machine learning practitioners after the seminal work of [23] and [35] on compressed sensing. These are, namely, the Lasso/Elastic Net regularization discussed in Section 5.4.1, the Group Lasso and one of its variants in which the groups are allowed to be overlapping (see Section 5.4.2 and Section 5.4.3 respectively), and finally the so-called Sparse PCA in Section 5.4.4. Clearly, since the emphasis remains on non-differentiable penalties leading to sparse solutions – in which currently there is a wide general interest – either optimal algorithms or non-optimal yet computationally very effective algorithms have already been proposed in the literature to solve these problems. However, recall that our specific aim here is not to compete directly with such algorithms, but instead to corroborate the initial statement about the generality and unifying properties of our framework.
Furthermore, we introduce two new algorithms for the specific problems of *multi-task learning* – in which several related tasks are considered simultaneously in order to take benefit by the sharing of information (sec. 5.4.5) – and *multiple kernel learning* – which is the process of finding an optimal kernel from a prescribed convex set of basis kernels. The simple form of the two algorithms makes it possible to verify immediately that both learning tasks are special instances of Group Lasso, in which the groups are defined task-wise and kernel-wise respectively.

It is important to note that, with respect to the general infinite dimensional setting we previously considered, we restrict to two particular settings that are of interest in machine learning and signal processing. The first is the case where we look for a function f that can be represented on a finite *dictionary of features* ψ_j, that is $f = \sum_{j=1}^M \beta_j \psi_j$ and the second one is the case when the function belongs to a Reproducing Kernel Hilbert space $\mathcal{H}$. In this latter we can still derive a finite dimensional representation for many regularization schemes using variation of the well known representer theorem and our approach gives a general way to find the coefficients in the representation.

5.4.1 Lasso and Elastic Net regularization

We are going to consider the following particular instance of the functional in (5.1)

$$\Phi_{\mu,\tau}^{(\ell_1 \ell_2)}(\beta) = \|\Psi\beta - y\|^2 + \mu \sum_{j=1}^{M} \beta_j^2 + 2\tau \sum_{j=1}^{M} w_j |\beta_j|, \tag{5.18}$$

where Ψ is a $n \times M$ matrix, β, y are the vectors of coefficients and measurements respectively, and $(w_j)_{j=1}^M$ are positive weights. The matrix Ψ can be thought as given by the features ψ_j in the dictionary evaluated at some points $x_1, \ldots, x_n$.

Minimization of the above functional is the so called elastic net, or ℓ_1-ℓ_2 regularization, proposed in [122], and reduces to the Lasso algorithm [106] if we set $\mu = 0$.

According to the notations introduced in the previous sections, the minimizer of (5.18) can be computed specializing Algorithm 3 to the case $F(\beta) = \|\Psi\beta - y\|^2 + \mu \sum_{j=1}^M \beta_j^2$ and $\mathcal{J}(\beta) = \sum_{j=1}^M w_j |\beta_j|$:

Algorithm 4 Lasso Algorithm

```
set $\beta^0 = 0$
for $p = 1, 2, \ldots,$ MAX_ITER do
    $$\beta^p = \mathbf{S}_{\frac{\tau}{\sigma}}((I - \frac{\mu}{\sigma})\beta^{p-1} + \frac{1}{\sigma}\Psi^T(y - \Psi\beta^{p-1})) \qquad (5.19)$$
end for
return $\beta^{\text{MAX\_ITER}}$
```

where $\mathbf{S}_{\tau/\sigma}$ acts componentwise as in (5.21) with $\lambda = \tau/\sigma$.

Note that the only delicate point in passing from equation (5.4) to equation (5.19) is the computation of $\pi_{(\lambda K)}$, since the argument of $\mathbf{S}_{\tau/\sigma}$ can be easily obtained by computing the derivative of $F(\beta)$. Applying Proposition 2 to $\mathcal{J}(\beta) = \sum_{j=1}^M w_j|\beta_j|$, with $\mathcal{H}_k = \mathbb{R}$ and $J_k\beta = w_k\beta_k$ $\forall k = 1, \ldots, M$, allows for solving (5.9) componentwise as

$$\bar{v}_j = \underset{|v|\leq 1}{\operatorname{argmin}}(\lambda w_j v_j - \beta_j)^2 = \operatorname{sign}(\beta_j)\min\left\{1, \frac{|\beta_j|}{\lambda w_j}\right\}. \qquad (5.20)$$

The non linear operation $(I - \pi_{\lambda K})$ acts therefore on each component as

$$[(I - \pi_{(\lambda K)})(\beta)]_j = \beta_j - \min\{|\beta_j|, \lambda w_j\}\operatorname{sign}(\beta_j) = \operatorname{sign}(\beta_j)(|\beta_j| - \lambda w_j)_+ = S_{\lambda w_v}(\beta_j) \qquad (5.21)$$

where $S_{\lambda w_j}$ is the well-known soft-thresholding operator.

From the above equation we see that, the iteration (5.19) with $\mu = 0$ leads to the iterated soft-thresholding proposed in [29] and derived also in [116]. In the general case, when $\mu > 0$, the iteration (5.19) becomes the damped iterated soft-thresholding proposed in [30]. In the former case, operator $\mathcal{T}_\sigma$ in (5.19) is not contractive but just non-expansive, since the Lipschitz constant in derived in Proposition 4 is not necessarily smaller than 1, unless Ψ has trivial null-space. Note that for sake of linearity we considered only ℓ_1-ℓ_2 regularization relative to finite dimensional spaces, $\mathcal{H}$. However the componentwise characterization of the projection in (5.20) can be shown to hold also in the infinite dimensional setting under suitable hypotheses (see [30] and references therein), and therefore our algorithm can be extended to encompass also this case.

5.4.2 Group LASSO

Here we consider a variation of the above algorithms where we assume the features to be disposed in *blocks*. This latter assumption was exploited in [117] to define the so called Group Lasso which amounts to minimizing

$$\Phi_{\mu,\tau}^{(grLasso)}(\beta) = \|\Psi\beta - y\|^2 + 2\tau\sum_{k=1}^{p} w_k\sqrt{\sum_{j\in\mathcal{I}_k}\beta_j^2}, \qquad (5.22)$$

where $(\psi_j)_{j\in\mathcal{I}_k}$ for $k = 1, \ldots, p$ is a block partition of the feature set $(\psi_j)_{j\in\mathcal{I}}$.

As in the previous case the only fact that needs explanation in specializing Algorithm 3 to this particular example is the computation of $\pi_{\lambda K}$. To this aim, note that applying Proposition 2, equation (5.9) can be decomposed componentwise as

$$\bar{v}_k = \underset{v \in \mathbb{R}^{|\mathcal{I}_k|},\ \|v\| \leq 1}{\operatorname{argmin}} \|\lambda w_k v - \beta_k\|^2_{\mathcal{H}} = \min\left\{1, \frac{||\beta^{(k)}||}{\lambda w_k}\right\} \frac{\beta^{(k)}}{||\beta^{(k)}||}$$

where $\bar{v} = (\bar{v}_1, \ldots, \bar{v}_p)$ with $\bar{v}_k \in \mathbb{R}^{|\mathcal{I}_k|}$, and $\beta^{(k)} \in \mathbb{R}^{|\mathcal{I}_k|}$ is the vector built with the components of $\beta \in \mathcal{H}$ corresponding to the elements $(\psi_j)_{j \in \mathcal{I}_k}$. The nonlinear operation $(I - \pi_{\lambda K})$ acts on each block as

$$[(I - \pi_{(\lambda K)})(\beta)]^{(k)} = \beta^{(k)} - \min\left\{\lambda w_k, ||\beta^{(k)}||\right\} \frac{\beta^{(k)}}{||\beta^{(k)}||} = \frac{\beta^{(k)}}{||\beta^{(k)}||}(||\beta^{(k)}|| - \lambda w_k)_+ \qquad (5.23)$$

The minimizer of (5.22) can hence be computed throug the iterative algorithm

Algorithm 5 Group Lasso Algorithm

```
set $\beta^0 = 0$
for $p = 1, 2, \ldots,$ MAX_ITER do
    $\beta^p = \tilde{\mathbf{S}}_{\frac{\tau}{\sigma}}(\beta^{p-1} + \frac{1}{\sigma}\Psi^T(y - \Psi\beta^{p-1}))$
end for
return $\beta^{\texttt{MAX\_ITER}}$
```

where $\tilde{\mathbf{S}}_{\tau/\sigma}$ acts blockwise as in (5.23) with $\lambda = \tau/\sigma$.

5.4.3 Composite Absolute Penalties

In [119], the authors introduce a novel highly flexible penalty, named Composite Absolute Penalty (CAP), based on possibly overlapping groups of features. The penalty is defined as:

$$\mathcal{J}(\beta) = \sum_{k=1}^{p} (\sum_{j \in \mathcal{I}_k} \beta_j^{\gamma_k})^{\frac{\gamma_0}{\gamma_k}},$$

where $(\psi_j)_{j \in \mathcal{I}_k}$ for $k = 1, \ldots, p$ is not necessarily a block partition of the feature set $(\psi_j)_{j \in \mathcal{I}}$.

This formulation allows to incorporate in the model not only groupings, but also hierarchical structures present within the features, for instance by setting $\mathcal{I}_k \subset \mathcal{I}_{k-1}$.

Clearly when $\gamma_0 = 1$, the CAP penalty is one-homogeneous and the solution can be computed through Algorithm 3. Moreover, if $\gamma_k = 2$ for all $k = 1, \ldots, p$, and F is the least square error, the CAP functional writes

$$\Phi^{(CAP)}_{\mu,\tau}(\beta) = \|\Psi\beta - y\|^2 + 2\tau \sum_{k=1}^{p} w_k \sqrt{\sum_{j \in \mathcal{I}_k} \beta_j^2}, \qquad (5.24)$$

where the penalty can be regarded as a particular case of (5.8), with $J^k : \mathbb{R}^d \to \mathbb{R}^{m_k}$ such that $\|J_k\beta\|^2 = \sum_{j=1}^{d} \beta_j^2 \mathbf{1}_{\mathcal{I}_k}(j)$, and $m_k = |\mathcal{I}_k|$. Note that, due to the overlapping structure of the features groups, the minimizer of (5.24) cannot be computed blockwise as in Algorithm 5, because the solution of the minimization problem (5.9) does not decouple on the blocks. However we can solve (5.9) through the iterative scheme (5.11), by identifying J with the matrix $(e_{j_1} \ldots e_{j_{m_k}})^T$ where $j_1, \ldots, j_{m_k} \in \mathcal{I}_k$, and building J as $(J_1^T \ldots J_k^T)^T$. We can then compute the minimizer of (5.24) through the iterative algorithm

Algorithm 6 CAP Algorithm

```
  set $\beta^0 = 0$
  for $p = 1, 2, \ldots,$ MAX_ITER_EXT do
    set $v^0 = 0$, $\tilde{\beta} = \beta^{p-1} + \frac{1}{\sigma}\Psi^T(y - \Psi\beta^{p-1})$
    for $q = 1, 2, \ldots,$ MAX_ITER_INT do
      $v_k^{q+1} = \frac{v_k^q - \eta J_k(J^T v^q - \sigma\tilde{\beta}/\tau)}{1 + \eta||J_k(J^T v^q - \sigma\tilde{\beta}/\tau)||}$
    end for
    $\beta^p = \tilde{\beta} - \frac{\tau}{\sigma} J^T v^{\texttt{MAX\_ITER\_INT}}$
  end for
  return $\beta^{\texttt{MAX\_ITER\_EXT}}$
```

5.4.4 Sparse PCA

Given a $n \times d$ matrix X, we aim at approximating it as $X \sim USV^T$, with U, S, and V $n \times s$, $s \times s$, $s \times d$ matrices respectively, such that the columns of V are sparse. In this problem which is known as Sparse Principal Component Analysis (SPCA), we want that the principal components of $X^T X$ separately depend on a small number of features. In this subsection we propose a new algorithm for solving the optimization problem proposed in [121] for SPCA, which amounts to minimizing the following functional

$$\Phi^{(SPCA)}_{\mu,\tau}(\alpha, \beta) = \sum_{i=1}^{n} \|X_i - \alpha\beta X_i\|^2 + \mu \sum_{j=1}^{d} |\beta_j^2| + \tau \sum_{j=1}^{d} |\beta_j| \quad \text{subject to } \alpha^T\alpha = I_s \tag{5.25}$$

with respect to the $s \times s$ α and β. The sparse principal components are then given by $V_j = \tilde{\beta}_j / ||\tilde{\beta}_j||$, where $\tilde{\beta}$ is the minimzer of (5.25) and $\tilde{\beta}_j$ is its i-th column. In this case we have that $F = \mathrm{Tr} X^T X + \sum_{j=1}^{s}(\beta_j^T (X^T X + \mu)\beta_j - 2\alpha_j^T X^T X \beta_j)$, so that $\nabla_{\beta_j} F = 2((X^T X + \mu)\beta_j - X^T X \alpha_j))$

5.4.5 Multitask Learning

Learning multiple tasks simultaneously has been shown to improve performance relative to learning each task independently, when the tasks are related in the sense that they all share a small set of features ([5, 16, 4, 87]).

In particular, given T tasks modelled as $f_t(x) = \sum_{j=1}^{d} \beta_{j,t} \psi_j(x)$ for $t = 1, \ldots, T$, according to

Algorithm 7 Sparse PCA Algorithm

```
set $\alpha^0 = 0$
for $q = 1, 2, \dots,$ M do

  for $j = 1, \dots, s$ do

    for $p = 1, 2, \dots,$ MAX_ITER_INT do
      set $\tilde{\beta}^0 = 0$
          $$\tilde{\beta}^p = \mathbf{S}_{\frac{\tau}{\sigma}}(\tilde{\beta}_j - \frac{1}{\sigma}((X^T X + \mu)\tilde{\beta}_j - X^T X \alpha_j^q))$$
    end for
    $\beta^q = \tilde{\beta}^{\texttt{MAX\_ITER\_INT}}$
  end for
  $\tilde{U}\tilde{D}\tilde{V}^T$ the singular value decomposition $X^T X \beta^q$
  $\alpha^q = \tilde{U}\tilde{V}^T$
end for
for $j = 1, \dots, s$ do
          $$V_j = \beta_j^{\texttt{M}} / ||\beta_j^{\texttt{M}}||$$
end for

return $V$
```

[87] the problem can be formalized as the minimization of the functional

$$\Phi_{\mu,\tau}^{(MT)}(\beta) = \sum_{t=1}^{T}\sum_{i=1}^{n_t}(\psi(x_{t,i})\beta_t - y_{t,i})^2 + 2\tau\sum_{j=1}^{d}\sqrt{\sum_{t=1}^{T}\beta_{t,j}^2}. \tag{5.26}$$

The last term combines the tasks and ensures that common features will be selected across them. Again functional (5.26) is a particular case of (5.1), and, defining

$$\beta = (\beta_1^T, \dots, \beta_T^T)^T$$

$$\Psi = \text{diag}(\Psi_1, \dots, \Psi_T), \qquad [\Psi_t]_{ij} = \psi_j(x_i).$$

its minimizer can be computed through the iterative algorithm 8, blue in which the component-wise soft-thresholding is substituted with a task-wise soft-thresholding $\tilde{\mathbf{S}}_\lambda$ acting simultaneously on the regression coefficients relative to the same variable in all the tasks.

5.4.6 Multiple kernel learning

Multiple kernel learning (MKL) [8] is the process of finding an optimal kernel from a prescribed convex set, $\mathcal{K}$, of basis kernels, $k_j(x, s)$, for learning a real-valued function by regularization. This problem has applications in kernel selection, and data fusion from heterogeneous data

Algorithm 8 Multi-Task Learning Algorithm

```
set $\beta^0 = 0$
for $p = 1, 2, \dots,$ MAX_ITER do
    $\beta^p = \tilde{\mathbf{S}}_{\frac{\tau}{\sigma}}(\beta^{p-1} + \frac{1}{\sigma}\Psi^T(y - \Psi\beta^{p-1}))$
end for
return $\beta^{\texttt{MAX\_ITER}}$
```

sources, and nonlinear feature selection [70]. In [82] the problem is formalized as the following double optimization

$$\inf_{k\in\mathcal{K}} \left\{ \inf_{f\in\mathcal{H}_k} Q(f) + \tau \|f\|_k^2 \right\} \tag{5.27}$$

where $\mathcal{H}_k$ is the RKHS with kernel k, $Q : \mathcal{K} \to \mathbb{R}_+$ is the empirical error. In the case where the set $\mathcal{K}$ is the convex hull of a finite number of kernels $k^{(1)}, \dots, k^{(M)}$, [81] show that the problem in (5.27) is equivalent to

$$\min \left\{ Q(f) + \tau \sum_{j=1}^M \|f_j\|_{k_j} : \; f = \sum_{j=1}^M f_j, f_j \in \mathcal{H}_{k_j} \right\}.$$

Interestingly, this amounts to finding a sparse representation of f w.r.t. the basis kernels.

When $\mathcal{H}_k = \mathcal{H}_{k_1} \otimes \cdots \otimes \mathcal{H}_{k_M}$ and $Q(f) = Q(f_1, \dots, f_M) = \frac{1}{n}\sum_{i=1}^n (\sum_{j=1}^M f_j(x_i) - y_i)^2$, the previous optimization problem becomes

$$f^* = \operatorname*{argmin}_{f=(f_1,\dots,f_M)} \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^M f_j(x_i) - y_i \right)^2 + 2\tau \sum_{j=1}^M \|f_j\|_{\mathcal{H}_{k_j}} \tag{5.28}$$

where f_j belongs to $\mathcal{H}_{k_j}$, given by the symmetric positive definite kernel k_j.

Though the space of functions is infinite dimensional the minimizer of the above functional (5.28) can be shown to have a finite representation. In fact, we can generalize the one-kernel representer theorem to obtain the following

Theorem 3. *The solution of the optimization problem* (5.28) *for* $\tau > 0$ *can be expressed as*

$$f^*(\cdot) = \left(\sum_{i=1}^n \alpha_{1,i} k(x_i, \cdot), \dots, \sum_{i=1}^n \alpha_{M,i} k(x_i, \cdot) \right).$$

Moreover, introducing the following notation

$\alpha = (\alpha_1, \dots, \alpha_M)$ with $\alpha_j = (\alpha_{j,1}, \dots, \alpha_{j,n})$,

$\mathbf{k}(x) = (\mathbf{k}_1(x), \dots, \mathbf{k}_M(x))^T$ with $\mathbf{k}_j(x) = (k_j(x_1, x), \dots, k_j(x_n, x))$,

$\mathrm{K} = (\mathrm{K}_1, \dots, \mathrm{K}_M)$ with $[\mathrm{K}_j]_{ii'} = k_j(x_i, x_i')$.

we can write the solution of (5.28) as $f^*(x) = (\alpha_1^T \mathbf{k}_1(x), \ldots, \alpha_M^T \mathbf{k}_M(x))$, so that we are allowed to restrict the search of the minimizer to the finite dimensional space spanned by $(\mathbf{k}_1, \ldots, \mathbf{k}_M)$. More precisely we can apply our general minimization procedure directly on the coefficients $(\alpha_1^T \mathbf{k}_1(x), \ldots, \alpha_M^T)$. We therefore consider the functional defined on $\mathbb{R}^{nM}$ as

$$\Phi_{\mu,\tau}^{MKL}(\alpha_1, \ldots, \alpha_M) = \frac{1}{n} \|\mathrm{K}\alpha - y\| + 2\tau \sum_{j=1}^{M} \sqrt{\alpha_j^T \mathrm{K}_j \alpha_j k}, \tag{5.29}$$

where y is the output vector. According to Theorem 1 the minimizer of $\Phi_{\mu,\tau}^{MKL}$ satisfies the fixed point equation

$$\mathcal{T}_\sigma(\alpha) = \left(I - \pi_{\tau/\sigma K}\right) \left(\left(\alpha - \frac{1}{\sigma n} \mathrm{K}^T \left(\mathrm{K}\alpha - y\right) \right) \right). \tag{5.30}$$

We now apply Proposition 2, noting that $J_j : \mathbb{R}^{nM} \to \mathbb{R}^n$ is the projection on the $j - th$ block of variables, the projection af an arbitrary vector $\alpha = (\alpha_1, \ldots, \alpha_M$ is $\pi_{\lambda K}(\alpha) = \lambda \bar{v}$ with

$$\bar{v} = \underset{v \in \mathbb{R}^{nM}, \|v_k\| \leq 1}{\operatorname{argmin}} \|\lambda v - \alpha\|_{\mathcal{H}}^2,$$

which can be computed block-wise as

$$\bar{v}_j = \min\left\{1, \frac{\|\alpha_j\|}{\lambda}\right\} \frac{\alpha_j}{\|\alpha_j\|} = \min\left\{1, \frac{\sqrt{\alpha_j^T \mathrm{K}_j \alpha_j}}{\lambda}\right\} \frac{\alpha_j}{\sqrt{\alpha_j^T \mathrm{K}_j \alpha_j}}.$$

The operation $(I - \pi_{\lambda K})(\alpha)$ is then given by

$$(I - \pi_{\lambda K})(\alpha) = \left(\frac{\alpha_1}{\sqrt{\alpha_1^T \mathrm{K}_1 \alpha_1}} (\sqrt{\alpha_1^T \mathrm{K}_1 \alpha_1} - \lambda)_+, \ldots, \frac{\alpha_M^T}{\sqrt{\alpha_M^T \mathrm{K}_M \alpha_M}} (\sqrt{\alpha_M^T \mathrm{K}_M \alpha_M} - \lambda)_+ \right) := \hat{\mathbf{S}}_\lambda(\mathrm{K}, \alpha)^T.$$

Finally the solution of (5.28) can be computed through the iterative algorithm

Algorithm 9 MKL Algorithm

```
set $\alpha^0 = 0$
for $p = 1, 2, \ldots,$ MAX_ITER do
    $$\alpha^p = \hat{\mathbf{S}}_{\tau/\sigma} \left( K, (\alpha^{p-1} - \frac{1}{\sigma n} \mathrm{K}^T (\mathrm{K}\alpha^{p-1} - y)) \right)$$
end for
return $\left(\alpha^{\texttt{MAX\_ITER}}\right)^T \mathbf{k}$.
```

5.4.7 Total Variation–based Image Denoising

As a first example we can consider the Total Variation Regularization for image denoising, which amounts to the minimization of the following functional:

$$\Phi^{(TV)}_{\mu,\tau}(f) = \|f - y\|^2 + 2\tau \sum_{i,j=1}^{n} \|[\nabla(f)]_{ij}\| \tag{5.31}$$

where y is a noisy $n \times n$ image, from which we aim at extracting the true image f, and ∇ is a linear discretization of the gradient operator. The minimization of $\Phi^{TV}_{\mu,\tau}$ can be easily recast in terms of (5.1). In fact, $f = \{f_{ij}\}_{i,j=1}^{n}$ so that $\mathcal{H} = \mathbb{R}^{n \times n}$, the operator A is the identity, $\mu = 0$ and $J_{ij}(f) = (\nabla f)_{ij} \in \mathbb{R}^2$.

Since $A = I$, as pointed out in Section 5.2, the solution is simply $f^* = y - \pi_{\tau K}(y)$, and the projection can be efficiently implemented through the iterative algorithm (5.11). If one approximates the ∇ operator by means of finite differences of neighbors pixels,

$$[(\nabla f)_{i,j}]_1 = \begin{cases} f_{i+1,j} - f_{i,j} & \text{if } i < n \\ 0 & \text{if } i = n \end{cases} \qquad [(\nabla f)_{i,j}]_2 = \begin{cases} f_{i,j+1} - f_{i,j} & \text{if } j < n \\ 0 & \text{if } j = n \end{cases}$$

this approach corresponds to the algorithm proposed by Chambolle in [26]. With this choice the adjoint of J is given by

$$(\nabla^T v)_{i,j} = \begin{cases} [v_{i-1,j}]_1 - [v_{i,j}]_1 & \text{if } \ 1 < i < n \\ -[v_{i,j}]_1 & \text{if } \ i = 1 \\ [v_{i-1,j}]_1 & \text{if } \ i = n \end{cases} + \begin{cases} [v_{i,j-1}]_2 - [v_{i,j}]_2 & \text{if } \ 1 < j < n \\ -[v_{i,j}]_2 & \text{if } \ j = 1 \\ [v_{i,j-1}]_2 & \text{if } \ j = n \end{cases} = -(\text{div } v)_{i,j}$$

and the minimizer of (5.31) can be computed through the iterative algorithm

Algorithm 10 Total Variation Algorithm

```
set $v^0 = 0$
for $p = 0, 1, \ldots,$ MAX_ITER do
    $$v^{q+1}_{i,j} = \frac{v^q_{i,j} + \eta(\nabla\,(\text{div } v^q + \sigma y/\tau))_{i,j}}{1 + \eta\,\|(\nabla\,(\text{div } v^q + \sigma y/\tau))_{i,j}\|_{\mathbb{R}^2}}.$$
end for
return $-\frac{\tau}{\sigma}\text{div } v^{\texttt{MAX\_ITER}}$
```

Part II

Gene Expression Microarray Data Analysis

Chapter 6

Finding structured gene signatures from microarray data

Post-genomic data generated by high-throughput technologies such as gene expression microarrays are increasingly used in medicine for diagnosis/prognosis, where the discovery of potential biomarkers is the main goal, and in molecular biology, in particular in functional genomics, where a major focus is the study of the cellular function for drug design. Therefore, differently from most standard learning problems, the extraction of relevant biological information from gene expression data – like disease subtype, survival time, or assessment of the gravity of an illness – requires not only to build an accurate predictor but also the identification of the list of the genes potentially involved in the process, genes which need to be further scrutinized by cross-checking the available knowledge or through additional quantification methods. Moreover, in a typical study, the size of the data set is less than a hundred, while the dimensionality of the data may be tens of thousands. As a consequence, feature selection, i.e., the identification of the gene signature actually involved in the process under study, is a formidable task for which classical statistics (designed to deal with large sets of data living in low-dimensional spaces) may not be well suited. Nontheless, a statistically robust feature selection tool is not sufficient to extract from gene expression data all the information that can be of interest for biologists and genetists. In fact, despite the availability of effective feature selection techniques, the biological interpretability of the selected gene lists is often a major problem. This is mainly due to the lack of structure in the selected gene signatures, and of complementary visualization tools. Therefore, another appealing goal of gene expression analysis is to devise statistical and visual tools able to interpret the results obtained by means of feature selection techniques and to understand their biological meaning. This requires some refined structure in the selected list of biomarkers, that can be used by biologists as a tool in the interpretation process, possibly leading to new biological hypotheses that can represent the starting point for further biological investigations.

In this chapter we deal with the problem of gene selection from microarray data, and propose an experimental protocol able to extract lists of relevant genes, organize them in modules of correlated genes ranked according to their prediction power, and efficiently visualize them. Starting off from the two-stage ℓ_1-ℓ_2 feature selection protocol described in Chapter 4, we propose an ad hoc agglomerative clustering technique able to refine such a nested output by explicitly identifying modules of correlated genes. We start from the first list which is minimal, meaning that the least number of discriminative genes is selected regardless of intra genes correlation. Such genes are used as centroids. Considering the subsequent lists with increasing size and comprising correlated genes we grow clusters of correlated genes around the obtained centroids. The main characteristic is that, differently from usual clustering approaches, the centroids (prototypes) are given and have a meaning in terms of classification ability. Another interesting information that can ease the interpretation process is provided by some ranking of the selected gene blocks. To this aim we propose a score based on the prediction power of the gene block in the supervised task under study. Furthermore, in order to enhance and easily interpret the obtained structure we propose two ad hoc visualization tools. First, we rearrange the covariance matrix according to the ordered blocks, and display it as an image. Second, we project the selected genes on the three most representative meta-patients. In this way we can extract and visualize a more structured genes signature, which captures and makes evident the correlation patterns in the data. This provides a richer model, that can be used to gain a better understanding of the genes function, possibly leading to new biological hypotheses.

In Section 1 we introduce the technology of gene expression microarrays, and present a brief survey of the most popular feature selection techniques applied to the analysis of high-throughput gene expression data.

In Section 2 we cast the gene selection problem as a standard variable selection task in the context of supervised learning. We than motivate the choice of ℓ_1-ℓ_2 regularization as a multivariate embedded selection algorithm which is at the same time correlation-aware and consistent.

In Section 3 we introduce our proposal for determining the genes modules from the two-stage output, for ranking them in terms of their prediction power and for visualizing them. We also challenge our method on synthetic data.

6.1 Gene expression microarray data

In every living organism there is a basic and fundamental working unit which we call the cell. Humans, like many other species, have trillions of cells (metazoa). Some organisms, such as yeast, have only one cell. We call these organisms protozoa. There are many types of cells (e.g. blood, skin, and nerve cells), but they all can be traced back to a single cell, the fertilized egg. Each cell of an organism contains a complete copy of the organism genome, i.e. the blue print for all cellular structures and activities in the body, which is transmitted from one generation to the next. This genetic material is encoded in DNA (deoxyribonucleic acid) molecules. The human genome is distributed along 22 pairs of autosomal chromosomes and one pair of sex chromosomes. A chromosome is made of compressed and entwined DNA (Figure 1). In each pair of chromosomes, one is paternally inherited and the other is maternally inherited. A DNA molecule is a double-stranded polymer structured in the form of a double-helix. It is composed of four basic molecular units called nucleotides. Each nucleotide is comprised of a phosphate group, a deoxyribose carbohydrate (sugar), and one of four nitrogen bases called adenine (A), guanine (G), cytosine (C), and thymine (T) (Figure 2). The two chains of the DNA are held together by hydrogen bonds between nitrogen bases (base-pairs). Base-pairing occurs only between G and C, or between A and T. The DNA molecule is organized into segments called "genes". An organism has the same genes in all its cells but they can be in different stages at different time moments. The genetic information stored into DNA may be transcribed into complementary RNA molecules which in turn may be translated into proteins (see Figure 6.1). Proteins are large molecules composed of one or more chains of amino acids. Proteins play a structural or functional role in all parts of an organism's body. As the result, for instance, proteins determine how foreign organisms and our body interact with each other (relevant to infectious diseases), or, for instance, how a body produces, or reacts to, cancerous cells. The DNA is responsible for the synthesis of proteins. The sequence in which base nucleotides appear in the DNA (base sequence) determines the order of amino acids that comprise a protein. As a result, these sequences determine the production level of hormones, enzymes, antibodies, etc. The DNA also determines the types of different cells, by using a mechanism called differential gene expression. Each cell contains a complete copy of the organism's genome. However, not all genes are expressed in all cells all the time. Differential gene expression determines where, when, and in what quantity each gene is expressed. This process produces different types of cells (skin, blood, nerve, etc.) using the same genome. Indeed, many complex human diseases and especially cancer are correlated with abnormal functionality at this level. For this reason, within molecular biology, the reaserch field known as functional genomics has gained an important role. Traditionally molecular biology has followed the so-called reductionist approach mostly concentrating on a study of a single or very few genes in any particular research project. With genomes being sequenced, in the last decade, this is now changing into the so-called systems approach where genomics analysis has shifted from focusing on one gene at a time to a more complex situation where the action and interaction of many thousands of genes can be measured simultaneously. Indeed, gene-expression microarrays, commonly called gene chips, make

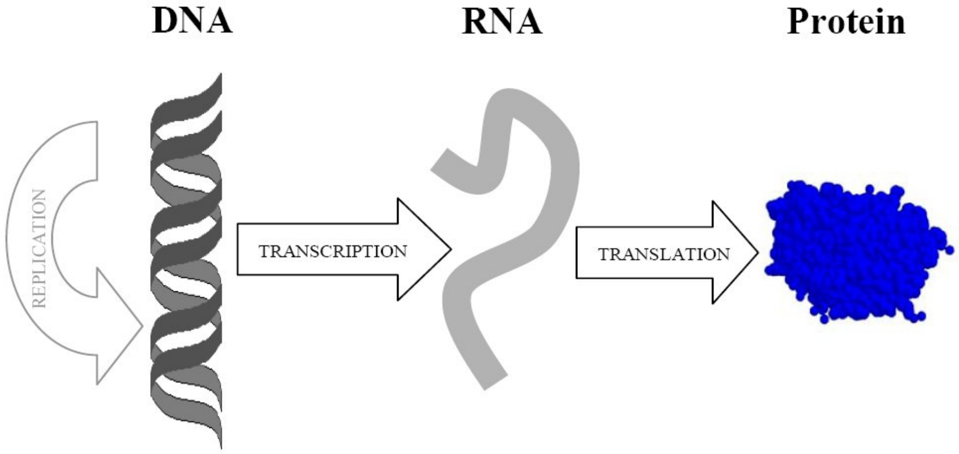


Figure 6.1: The central dogma of molecular biology. When a gene is expressed, it is first transcribed into an RNA sequence, and the RNA is then translated into a protein, a sequence of amino acids.

it possible to simultaneously measure the rate at which a cell or tissue is expressing – translating into a protein – each of its thousands of genes, generating a global picture of the cellular function.

6.1.1 Microarray technology

In microarrays single strands of complementary DNA (RNA) for the genes of interest - which can be many thousands - are immobilized on spots arranged in a grid on a support which will typically be a glass slide, a quartz wafer, or a nylon membrane (see Figure 6.2). From a sample of interest, e.g. a tumor biopsy, the mRNA is extracted, labeled and hybridized to the array. Probe-target hybridization is then detected and quantified by fluorescence-based detection of fluorophore-labeled targets. This yields an intensity value that should be correlated to the abundance of the corresponding RNA transcript in the sample.

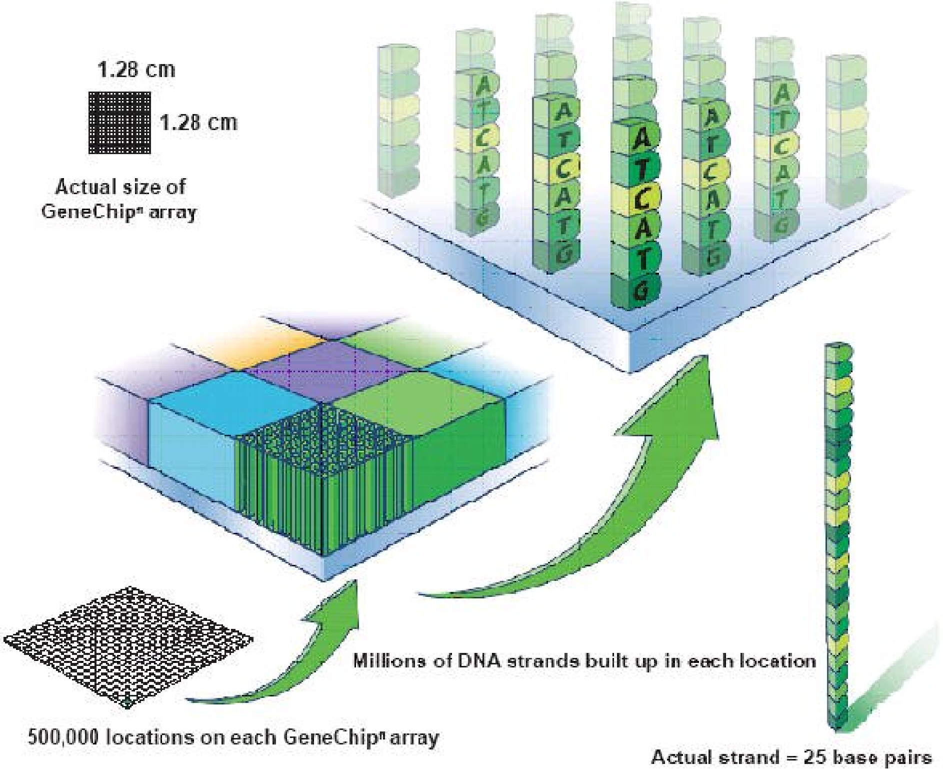

1.28 cm
1.28 cm
Actual size of GeneChip® array
Millions of DNA strands built up in each location
500,000 locations on each GeneChip® array
Actual strand = 25 base pairs
ATCATG

Figure 6.2:

The two major platforms for microarrays in common use today are *spotted* and *oligonucleotide* microarrays. In *spotted* arrays the probes are mechanically deposited onto modified glass slides by contact or ink jet printing. The resulting "grid" of probes represents the nucleic acid profiles of the prepared probes and is ready to receive complementary cDNA or cRNA "targets" derived from experimental or clinical samples. These arrays may be easily customized for each experiment, because researchers can choose the probes and printing locations on the arrays, synthesize the probes in their own lab (or collaborating facility), and spot the arrays. In *oligonucleotide*, or *in situ* microarrays, the probes are short sequences designed to match parts of the sequence of known or predicted open reading frames. Oligonucleotide arrays are produced by printing short oligonucleotide sequences designed to represent a single gene or family of gene splice-variants by synthesizing this sequence directly onto the array surface instead of depositing intact sequences. Sequences may be longer (60-mer probes such as the *Agilent* design) or shorter (25-mer probes produced by *Affymetrix*) depending on the desired purpose; longer probes are more specific to individual target genes, shorter probes may be spotted in higher density across the array and are cheaper to manufacture. For a comparative review of the two platforms, we refer to [57].

While the former technique provides a relatively low-cost array that may be customized for each study, in many exploratory gene expression studies commercial oligonucleotide microarrays are preferred. Therefore, microarrays high-troughput technology, more than any other, has contributed to generate an amazing amount of data that over the years have been piled up. Despite the huge amount of potential information enclosed in microarray measurements, the amount of data this new technology produces is more than one can manually analyze. The extraction of biologically meaningful information from these data, can hence be accomplished only through a multidisciplinary approach to data analysis in which biological knowledge are integrated by bioinformatic skills and advanced statistical applications. As a consequence, the need for automated analysis of microarray data offers an opportunity for machine learning to have a significant impact on biology and medicine.

6.1.2 State-of-the-art on gene selection for microarray data

The analysis of microarray gene expression data has gained a central role in the process of understanding biological processes. The comprehensive snapshot of biological activity provided by microarray measurements can be used to infer regulatory pathways in cells, identify novel targets for drug design, improve the diagnosis, prognosis, and treatment planning for those suffering from disease.

Many expression studies have so far focused on devising methods to cluster genes by similarities in expression profiles. This is in order to determine the proteins that are expressed together under different cellular conditions. Briefly, the most common methods are hierarchical clustering, self-organising maps, and K-means clustering. Hierarchical methods originally derived from algorithms to construct phylogenetic trees, and group genes in a bottom-up fashion; genes with the most similar expression profiles are clustered first, and those with more diverse profiles are included iteratively [41, 1]. In contrast, the self-organising map [102, 107] and K-means methods [103] employ a top-down approach in which the user pre-defines the number of clusters for the dataset. The clusters are initially assigned randomly, and the genes are regrouped iteratively until they are optimally clustered.

More complex relationships have also been assessed. Conventional wisdom is that gene products that interact with each other are more likely to have similar expression profiles than if they do not [42]. One of the main driving forces behind expression analysis has been to analyze cancerous cell lines. In general, it has been shown that different cell lines can be distinguished on the basis of their expression profiles. The basis for their physiological differences were apparent in the expression of specific genes; for example, expression levels of gene products necessary for progression through the cell cycle correlated well with variations in cell proliferation rate. Comparative analysis can be extended to tumour cells, in which the underlying causes of cancer can be uncovered by pinpointing areas of biological variations compared to normal cells. For example in breast cancer, genes related to cell proliferation and the IFN-regulated signal transduction pathway were found to be upregulated [88]. One of the difficulties in cancer treatment has been to target specific therapies to pathogenetically distinct tumour types, in order to maximize efficacy and minimize toxicity. Thus, improvements in cancer classifications have been central to advances in cancer treatment. Although the distinction between different forms of cancer – for example subclasses of acute leukemia – has been well established, it is still not possible to establish a clinical diagnosis on the basis of a single test. In a recent study, acute myeloid leukemia and acute lymphoblastic leukemia were successfully distinguished based on the expression profiles of these cells [54]. As the approach does not require prior biological knowledge of the diseases, it may provide a generic strategy for classifying all types of cancer. Another interest problem related to gene expression analysis is concerned with the prediction of the tumor evolution – i.e. aggressiveness and risk of relapse –.

The problems described above can be devided in two main classes: classification problems such as cell line discrimination and cancer subtyping , and regression problems, where the goal is to determine a quantitative parameter, such as the cell proliferation of the tumor aggressiveness.

In both cases machine learning represents a valuable tool. In particular, where prediction represents the main goal, many learning strategies have been proposed, and appear to lead to efficient generalization. In the context of gene expression data, an extensive survey on the applications of support vector machines to genomics can be found in [110]. Other standard learning techniques, such as boosting [50], have also been applied with success to DNA data analysis. These approaches are especially important in the potential clinical application of microarrays where the power of the technology is in its ability to use somewhat imprecise composite patterns of expression. However, most of these methods return a classifier that depends on a large number of features and are not naturally suited to perform gene selection. In this context a major goal is thus to provide reliable gene selection techniques, that work in the typical *-omics* scenario of a small number of samples represented in a high dimensional space and are able to capture the complex interactions among genes by selecting the variables most relevant to characterize a given task.

In classification tasks, a typical fundamental question is: *how do expression levels of specific genes or gene-sequences differ in a control group versus a treatment group?* One of the simple methods attempting to answer the above question is called the fold approach. In this approach, if the average expression level of a gene has changed by a predetermined number of folds, that gene expression is declared to have been changed due to the treatment. In many studies, a 2-fold technique is used (rather than 3-fold or 4-fold), in which the average expression level has to change to at least two folds of its initial level in order for it to be classified as changed. The drawback of this method is that it is unlikely that it reveals the desired correlation between expression data and function, as a predetermined factor of 2 (or 3 or 4) has different significance depending on the expression levels of various genes. A further drawback is that this method only compares the expression level of the gene under question to determine whether it has been turned on or off. A better, and more biologically relevant method of analysis, would be to consider expression patterns of a set related genes to determine the on or off state of the gene currently under observation. The fold approach does not allow this type of analysis. Similar to the fold approach, hypothesis testing, such as t-test, is another simple method applied in gene expression analysis. The fundamental problem with these tests is that they require repeated treatment and control experiments, which is extremely costly. As a result, a small number of experimental repetition could affect the reliability of the mean-based approach. Furthermore, when applying hypothesis testing to gene expression data, it is important to take into account the problem of multiple testing in order to avoid generating many false leads (sometimes referred to as "false discoveries"). For example, if 10,000 genes were tested, we would expect 500 genes to be falsely declared as significantly different between the classes at p-value lower than 0.05, even if there were no real differences. Various multiple comparisons adjustment procedures have been applied to microarray data for purposes of controlling the number or proportion of false discoveries. We refer to [39] for a detailed survey of hypothesis tests, and multiple test corrections, and to [54, 75] for their application to gene selection from microarray data.

While folding and hypothesis testing are examples of filter methods applied to gene expression data analysis, among wrappers, *gene shaving* [60] and algorithms derived from Support Vector Machines (SVM) [110] are the methods that have been most commonly applied to the problem of gene selection from microarray data. Gene Shaving is a method for clustering groups of similarly behaving genes whose changes in expression are most tightly linked to observed biological changes. The basic method is similar to principal components analysis with a sequential twist: a canonical "gene vector" is identified based on the eigenvectors, and the genes are ranked according to their agreement with this vector. The worst fitting are then "shaved off" and a new canonical vector is identified and fit. Among algorithms derived from Support Vector Machines, recursive feature elimination (RFE) procedure of [56], which iteratively selects smaller and smaller sets of genes and trains SVM, is the most popular. RFE can only be applied with linear SVM, which is nevertheless not a limitation as long as many features remain, and works as follows. Starting from the full set of genes, a linear SVM is trained and the genes with the smallest weights in the resulting linear discrimination function are eliminated. The procedure is then repeated iteratively starting from the set of remaining genes, and stops when a desired number of genes is reached. A variation of this method is the entropy-based recursive feature elimination for SVMs [52]. This is a non-parametric procedure for gene ranking, which accelerates – without reducing accuracy – the standard recursive feature elimination (RFE) method for SVMs.

Differently from wrappers and filters, where variable selection and training are two separate processes, embedded methods present the advantage of incorporating gene selection within the construction of the classifier or regression model. Besides decision trees [21] and boosting methods such as the popular Adaboost [49], an appealing new trend has emerged recently namely the use of penalized methods in genomics or proteomics. These methods consist in the minimization of a well-defined objective function to which a penalty term is added in order to avoid "overfitting", i.e. to provide some form of "regularization" – or equivalently an implicit reduction of the dimensionality of the feature space. A variety of such methods have been proposed in the recent literature and differ by the choice of the objective function and of the penalty term; for a recent overview, see e.g. [97],[76] and the references therein.

Particularly interesting are penalties which allow to enforce sparsity of the model, namely to perform automatic gene selection by assigning truly zero weights to all but a small number of selected genes. The most famous example are the ℓ^1-type penalties used in the so-called LASSO regression, a name coined by [106] as an acronym for "Least Absolute Shrinkage and Selection Operator". The use of LASSO for genomics is also advocated e.g. in the recent papers by [53] and [96]. However, as already pointed out in Chapter 4, a drawback of LASSO regression in the presence of groups of correlated genes is that the method is not able to identify all members of the group. For this reason "elastic net" [122], or ℓ^1-ℓ^2 regularization, which will be described in details in the rest of the chapter, represents an efficient alternative to the LASSO method able to overcome such drawback and to identify groups of correlated genes.

6.2 Embedded Gene Selection via ℓ^1-ℓ^2 optimization

In the context of gene selection, even though wrappers have been estensively employed (see [52], and references therein), their greediness makes it preferable to invoke embedded methods for gene selection. Before describing our embedded approach to the gene selection problem, let us recast it as a standard feature selection task in the learning-from-examples framework.

We assume we are given a set of n examples as input/output pairs. We denote the inputs with $x_i \in \mathcal{X} = \mathbb{R}^d$, $i = 1, ..., n$; in our case the components of the vector x_i are the expressions of the d probe sets synthesized on the chip for each patient i. We note that n may be about 100 or 1000 times smaller than d. The outputs, or corresponding responses, are denoted with $y_i \in \mathcal{Y}$ and can be either a discrete class label in classification problems (e.g. discriminating between disease subtypes), or a continuous real variable in regression problems (e.g. a measurement of some biological parameter, survival time, or assessment of the gravity of the illness). The problem we face is to find which of the d components are needed to predict the response y as accurately as possible from any given input x. In our case the model cardinality is known to be much smaller than d, though the complexity of gene regulatory networks makes it difficult to determine the number of genes actually involved in the process.

We restrict our attention to linear functions, or equivalently to vectors $\beta \in \mathbb{R}^d$, modelling the relation between x and y as $y = \beta \cdot x$. As customary in learning theory, the given examples pairs are assumed to be drawn i.i.d. from a fixed but unknown probability density $p(x, y)$ with $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Therefore, if the risk of predicting $\beta \cdot x$ instead of y is measured by $(\beta \cdot x - y)^2$, the expected risk of a given model β, in the least-squares sense, is given by

$$\mathcal{E}[\beta] = \int_{\mathcal{X}\times\mathcal{Y}} (y - \beta \cdot x)^2 \, p(x, y) \, \mathrm{d}x \, \mathrm{d}y \,. \tag{6.1}$$

The goal is to determine a *sparse* model β^*, i.e. a model of cardinality much smaller than d – that is, a vector β^* with only s entries different from zero (with $s << d$) – for which the expected risk, $\mathcal{E}[\beta^*]$, takes on a small value. We recall that the components of the model vector are called regression coefficients or weights.

Before illustrating our approach, let us add a few considerations on the desirable properties of a gene selection algorithm.

Multivariate. Despite the popularity of the filter approach, in particular of statistical tests, in the analysis of microarray data, it is quite naive to believe that gene interactions are restricted to univariate dependences between a single gene expression and the output, since these methods discard potentially relevant information concerning the combined effect of groups of two, three or even many genes together. Therefore it is often desired to develop a multivariate predictor for classification and regression tasks taking into account at least linear interactions of multiple genes. For example, there may be a number of gene markers whose collective behavior may predict with substantial accuracy whether a tumor will respond to a particular chemotherapeutic agent. This situation is sketched in a simplified way in Figure 6.3, where we can think of class 1 and class 2 as normal and tumor samples. In this case upregulation of either gene 1 or gene 2 can combine to bring about tumor outset. Clearly univariate analysis will fail. The question is then how to best mathematically combine the gene expression measurements into a single function that can delineate the tumor type, survival or other biological parameters.

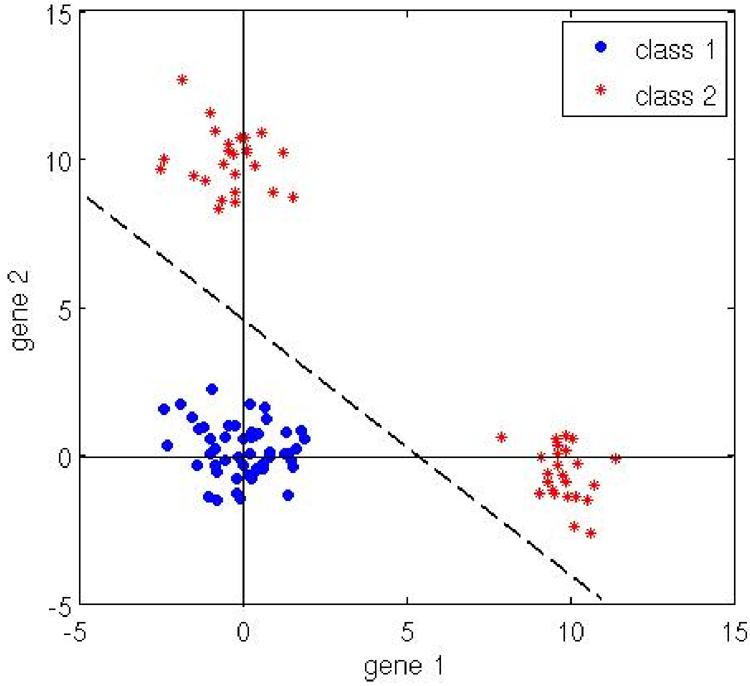


Figure 6.3: Multivariate dependance

Correlation-aware. A good gene selector should avoid discarding correlated variables. In fact, while accurate prediction can be achieved by means of many different techniques, gene identification, due to gene correlation and the limited number of available samples, is a much more elusive problem. Small changes in the expression values often produce different gene lists, and solutions which are both sparse and stable are difficult to obtain. As a consequence, despite many favorable properties, most sparsity based methods have shortcomings in the presence of correlated genes. *Classification performance does not change if we select one or more relevant but correlated genes*, and the selection among correlated genes is done in an uncontrolled way often depending on the particular implementation. Therefore, an important requirement is that, when one variable is considered relevant to the problem, its correlated variables should be considered relevant as well.

Consistent. From the statistical viewpoint a minimal requirement is asymptotic consistency of the algorithm, ensuring that results improve as we get more data, and eventually the best possible estimator is reached when enough data is available. The second consideration is extremely important in the analysis of gene expression data, where a large number of heuristics have been used to tackle the gene selection problem, because, due to the limited number of training samples, the learning phase is at risk of over-fitting, and generalization is therefore a basic requirement.

Given these important requirements, we propose the two-stage ℓ_1-ℓ_2 regularization method, described in Chapter 4, as an efficient and robust tool for selecting relevant genes from microarray data. Indeed, this method satisfies the three requirements described above. Moreover, in several problems the ability of returning nested list of relevant genes is often regarded as the most precious information for further investigation based on biological knowledge and subsequent experimental validation. In fact, the nested structure of the selected gene lists allows to choose the desired level of complexity to be used in the biological investigation of the underlying phenomenon. In other words we can choose how many genes we want to consider for further studies, maintaining all the information extracted from the data. For example, when interested in finding a set of biomarkers to be used on large scale diagnostic tests, one might prefer a small panel of significant genes due to time, cost and resources limitations, and the minimal gene list (the one obtained with $\mu = \mu_1$) can thus be employed. On the other hand when the major goal is the comprehension of the entire cell response to an external factor, such as a particular drug treatment, the maximal list (μ max) is preferable.

6.3 Towards System Biology

The main idea of this section is to take a step further with respect to ℓ^1-ℓ^2 regularization and propose a robust statistical analysis framework able to extract gene lists that are structured in modules of correlated genes ranked according to their discriminative power. Moreover, thanks to ad hoc visualization tools, the obtained signature can be effectively visualized making the search of interesting biological patterns easier. This step could be of great help in the comprehension of the cellular behavior in that the obtained gene modules could reflect the true structure of the functional families of genes. Indeed genes that are functionally correlated are also likely to present expression values with high Pearson correlation across different samples.

In Subsection 6.3.1 we show how to apply an appropriate ad hoc clustering technique to extract the modules of correlated genes from the output of the two-stage procedure described in Section

4.3. In Subsection 6.3.2 we then discuss how the obtained modules can be ranked according to their discriminative power. Finally, in Subsection 6.3.3 we present two customized visualization tools that allow to better interpret and appreciate the obtained results.

6.3.1 Structered Clustering of two-stage ℓ_1-ℓ_2 output

The two-stage approach described in Section 4.3 has proved to be an efficient technique for feature selection.Moreover, we have described how the nested structure of the selected gene lists allows to choose the desired level of complexity to be used in the investigation of the underlying biological process. Despite this appealing property, the raw structure of the gene lists does not provide any understanding on how to group correlated genes. All we have is a set of lists ordered according to their size. Starting from the first minimal list of genes the following lists include more correlated genes. Therefore, we now present a procedure that allows to go one step further towards the biological interpretability of the selected gene signatures.

The idea is to build blocks of correlated genes using a variation of well known agglomerative clustering techniques. The technique is based on the Pearson distance:

$$d(X_{j_1}, X_{j_2}) = \frac{corr(X_{j_1}, X_{j_2})}{\sqrt{var(X_{j_1})var(X_{j_2})}}$$

evaluating the (normalized) correlation between the j_1-th and the j_2-th columns (genes) of the data matrix X. By definition of minimality, the B genes in the minimal list (the one obtained with $\mu = \mu_1$) can be assumed to be independent. Therefore the structured gene signature will be constitued by B blocks. Each gene p_b in the minimal list will be used as a *prototype* for the b-th block. For each gene in the maximal list, correlation is calculated with each prototype and hence the gene is assigned to the block (prototype) associated to higher correlation. In this way we populate the blocks corresponding to the prototype genes in the minimal list with all the genes coming from the maximal list and encompassing all its sublists. Within each block we sort its genes according to the order of appearance while performing the two stage elastic net selection, for increasing $\mu = \mu_1, \dots, \mu_{\max}$. In block b, the prototype p_b, selected for μ_1, will be ranked first, its correlated genes appearing at μ_2 will be ranked second (with ties) and so on.

Note that, the main characteristic of the described procedure is that, differently from usual clustering approaches, the centroids (prototypes) are fixed in advance and have a meaning in terms of classification ability. On the other hand, in clustering algorithms like k-means, initial centroids are given just as starting point, and can be completely modified by the evolution of the algorithm.

6.3.2 Ranking Criterion

In the study of the cellular behavior, not all the selected genes, or gene modules, are equally relevant to the underlying phenomenon. As a consequence, in most functional genomics studies, it might be preferable to sort the gene lists in order to provide some order of relevance suggesting which genes, or gene modules, shall be analyzed first. We thus propose a criterion for ranking the gene blocks according to their prediction power with respect to the supervised problem under study. Since the estimator $y \sim f(x)$ is a weighted sum of the genes expressions, $f(x) = \sum_{j=1}^{d} X_j \beta_j$, we can define the score, s_b, for block b as the contribution given to $f(x)$ by the prototype of block b:

$$s_b = \left\| X_b^{(1)} \beta_b^{(1)} \right\|$$

where $X^{(1)}$ is the gene expression matrix resctricted to the genes selected with the lowest value of μ, and $\beta^{(1)}$ is the corresponding optimal weight vector.

6.3.3 Visualization of the Structered Clustering

Biological data analysis and visualisation have traditionally been approached as independent problems. Relatively little attention has been given to the integration and visualisation of information and models. However, the integration of these areas facilitates a deeper understanding of problems at a systemic level. Therefore, in order to interpret and appreciate the results obtained with the agglomerative clustering technique proposed in Subsection 6.3.1, we propose two simple visualization tools.

In the first method, we first restrict the gene expression matrix to the probe sets belonging to the blocks union. We then rearrange its columns in order to emphasize both the blocks and the layered structure, and display its correlation matrix as an image: genes belonging to the same groups are drawn close to each other, and thicks lines separate each block; in addition, within each module, the gene in the upper left corner has to be identified with the prototype or first layer gene, whereas the genes selected with increasing value of μ follow, separated by a thinner line.

In the second visualization, we project the genes appearing in the blocks union on the most representative *meta-patient*. Such meta-patient is identified with the 3D space spanned by the first three left eigenvectors of the normalized expression matrix X_{ij} restricted to the genes which belong to the blocks union. Clearly highly correlated genes cluster together in the 3-dimensional space, while collinear genes present perfect overlap.

6.4 Toy Data

In order to test our approach in a controlled setting, we applied the two stage elastic net regularization, followed by the nesting-clustering technique on a toy example, where we exactly know which are the relevant or correlated features. The proposed toy problems are close to real data conditions, e.g. dependence on more than one variable and correlation, though in a lower dimensional setting. A set of $n = 30$ toy-patients are drawn from $\mathbb{R}^d$ with $d = 50$ in the following way:

$$\begin{array}{lll} x_1, x_{11}, x_{21}, x_{31}, \ldots, x_{50} & \text{i.i.d. from } \{-0.5, 0.5\}, \\ x_{1+i} & \sim s_i \cdot x_1 \; + \; \sigma_1 \varepsilon & \text{for } i = 1, \ldots, 9, \\ x_{11+i} & \sim s_i \cdot x_{11} + \; \sigma_2 \varepsilon & \text{for } i = 1, \ldots, 9, \\ x_{21+i} & \sim s_i \cdot x_{21} + \; \sigma_3 \varepsilon & \text{for } i = 1, \ldots, 9, \end{array}$$

where $\varepsilon \sim N(0,1)$. A combination of features x_1, x_{11} and x_{21} separates the toy-patients in two classes (multivariate model) according to the rule:

$$\begin{array}{rcl} \Pr\left(y = +sign(X \underline{+} \varepsilon) \right) & = & p \\ \Pr\left(y = -sign(X \underline{+} \varepsilon) \right) & = & 1 - p \end{array}$$

where $\beta = (1, 1, 1, 0, \ldots, 0)$.

The family of toy problems described above consider three discriminating groups each containing 10 correlated features $(\{x_1, \ldots, x_{10}\}, \{x_{11}, \ldots, x_{20}\}, \{x_{21}, \ldots, x_{30}\})$, while features $x_{31}, \ldots, x_{50}$ are uninformative. By applying our technique we aim at selecting and clustering such relevant blocks. We now examine the performance of our technique on three toy problems which differ in the correlation parameters $\sigma_1, \sigma_2, \sigma_3$, scaling factors $s_i, i = 1 \ldots, 9$ and Bayes risk p. We state beforehand that the selection step allows to achieve optimal prediction performance on all toy problems taken under consideration. Being interested in the second step, in the following we analyze in details the results on clustering and visualization.

6.4.1 Toy Problem 1

In this toy problem, genes belonging to the same block, have comparable expressions ($s_i = 1,\ i = 1 \ldots, 9$), and high correlation with either x_1, x_{11} or x_{21} ($\sigma_1 = \sigma_2 = \sigma_3 \sim 0.1$). The two classes are perfectly separated ($p = 0$).

As shown in Figure 6.4 (top), the first stage of the $\ell - 1$-ℓ_2 regularization selects 4 minimal features instead of 3: x_3, x_{19}, x_{26} and x_{24}. Note that the last two features belong to the same block and therefore are highly correlated; however, due to the non-perfect correlation, they are both selected at the first stage. By increasing the ℓ_2 parameter all features from block 1 and 2, and 6 of the 10 features from block 3 are subsequently selected. Clearly from Figure 6.4 (top), the nesting-clustering technique succeeds in correctly assigning each feature to its corresponding block. According to the data generation rule, Figure 6.4 (bottom) shows that clusters 1, 2 and 3 are correctly far apart when projected on the 3D meta-patients, whereas block 4 clearly overlaps block 3.

6.4.2 Toy Problem 2

We now add a small amount of noise to the classification problem by raising the error probability, p, to 0.01. The other parameters are left unchanged as in Toy problem 1.

As in the previous case the first stage of the elastic net algorithm selects 4 minimal features instead of 3: x_{14}, x_7, x_{29} and x_{23}, where, again features x_{29} and x_{23} are correlated. With higher

μ almost all the features from the relevant blocks are recovered, and again the nesting-clustering technique succedes in correctly grouping the features. In Figure 6.5 the same behavior as in Toy Problem 1 is shown

6.4.3 Toy Problem 3

In the last toy problem, features in the three relevant blocks are rescaled replicates ($\sigma_1 = \sigma_2 = \sigma_3 = 0$) of either x_1, x_{11} or x_{21}; this makes the features in the same block exactly correlated however the non-unit scaling factors ($s_i = 1 - i/10,\ i = 1 \ldots, 9$), make their expression values non comparable. As in the first problem, classes are perfectly separated ($\sigma_0 = 0$).

In this case, all the correlated features belonging to the same block bring the same amount of information, being their correlation equal to 1. In fact, the family of solutions

$$\left\{ f(x) = \sum_{j=1}^{30} x_j \beta_j \ : \ \beta_1 + \sum_{j=1}^{9} \beta_{1+j} s_j = \beta_1^*, \ \ \beta_{11} + \sum_{j=1}^{9} \beta_{11+j} s_j = \beta_2^*, \ \ \beta_{21} + \sum_{j=1}^{9} \beta_{21+j} s_j = \beta_3^* \right\}$$

are exactly equivalent in terms of prediction performance. As a consequence, being $|s_i| < 1$, the ℓ_1 penalty favours solution $x_1\beta_1^* + x_{11}\beta_2^* + x_{21}\beta_3^*$, as the one having lowest ℓ_1 norm. The trade-off between the ℓ_1 and ℓ_2 norms, has the effect of recovering more and more features, as the ℓ_2 parameter increases. As shown in Figure 6.6, at stage I the algorithm selects the 3 relevant features with highest expression, 1, 11 and 21, and two noisy features, 36, 37. The second stage progressively includes almost all the corelated features according to their scaling factor, and the clustering algorithm correctly groups them.

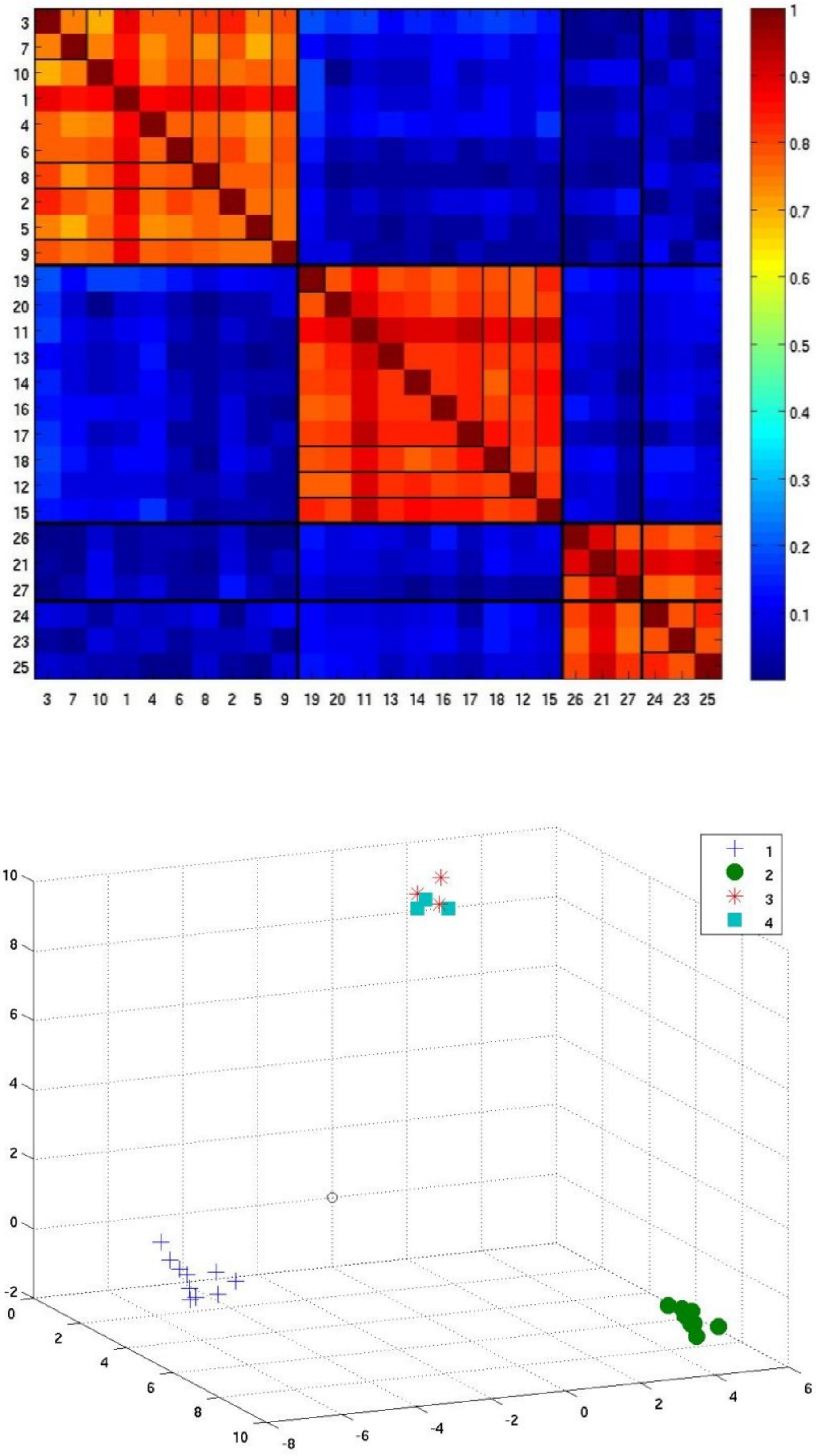


Figure 6.4: Correlation and Clustering in Toy Problem 1 .

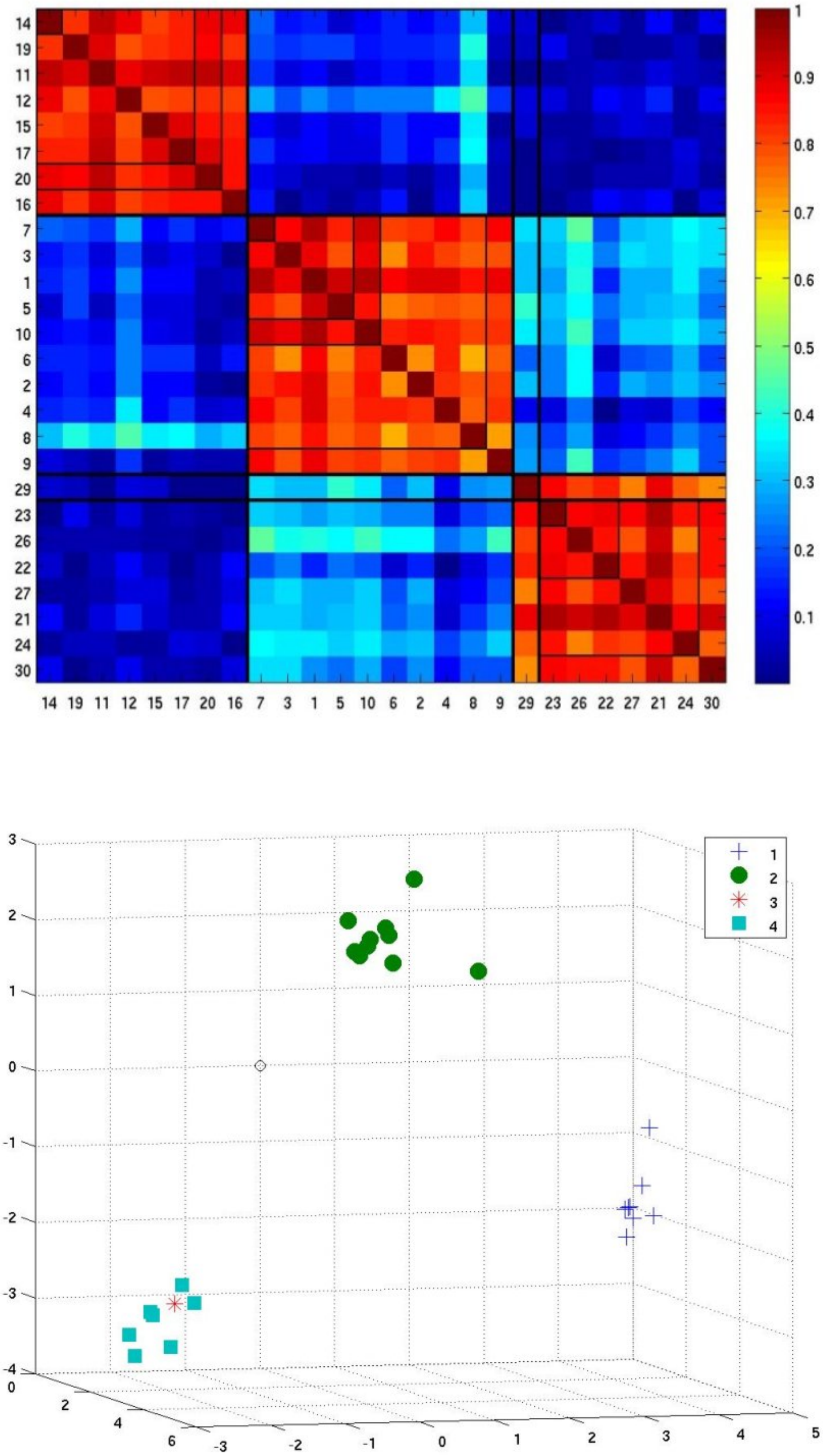


Figure 6.5: Correlation and Clustering in Toy Problem 2.

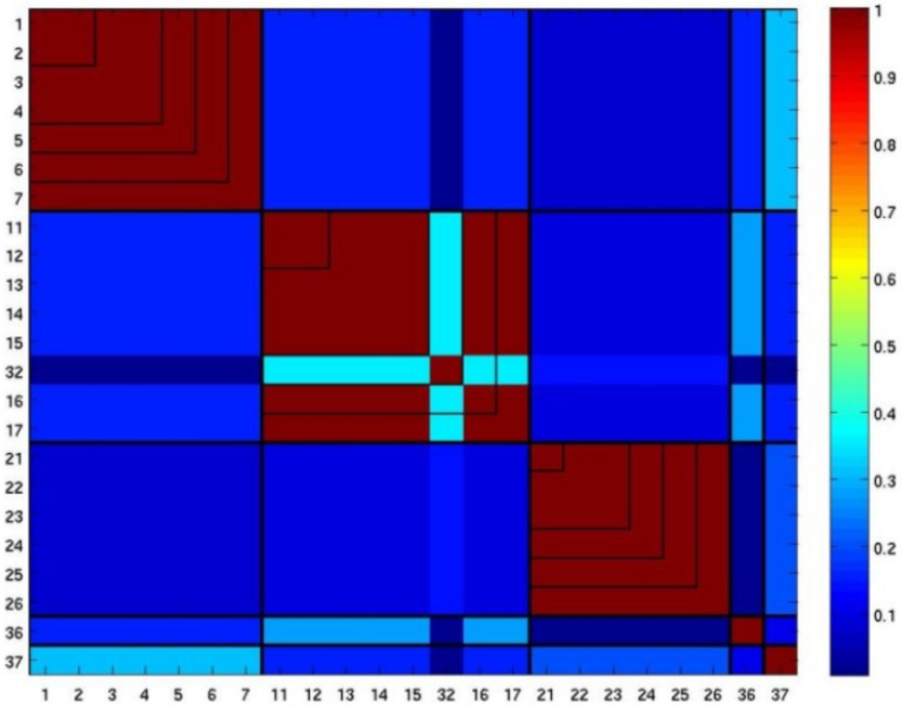


Figure 6.6: Correlation on replicated Toy Problem 3

Chapter 7

Real data experiments

Signatures represent a defined panel of genes that characterize a biological response or a pathological process. The definition of signatures is instrumental for defining the relevance of given biological responses to the progression of disease as in the case of tumors. In the previous chapter we have introduced our procedure for extracting significant signatures from high-throughput data. On synthetic data the proposed procedure proved to be highly effective in identifying the relevant genes. In this chapter we therefore examine the performance of the proposed experimental protocol on benchmark microarray data sets, and then apply it to the analyis of new in vitro cell lines arrays.

In Section 1 we tested the two-stage approach on three data sets previously considered in microarray studies concerning classification tasks. The data sets under consideration comprise one data set of in vitro samples, and three sets of data from in vivo tissue of three different diseases.

In Section 2 we challenge our experimental protocol on a new medical trial, in collaboration with the children hospital Giannina Gaslini in Genova. The goal of this experiment is to extract a signature for hypoxia, based on the gene expression profile of a cohort of in vitro neuroblastoma cell lines.

7.1 Benchmark data sets

7.1.1 Cell-culture microarray data

We now analyze the RAS data set used in [18] and available on line at `http://data.genome.duke.edu/oncogene.php`. In [18] human primary mammary epithelial cell cultures (HMECs) were used to develop a series of pathway signatures related to different oncogenes. In short, cells were infected with different adenoviruses for eighteen hours and signatures were extracted as the set of genes most correlated to the classification of HMEC samples into oncogene-activated versus control. In order to test our method for gene selection, we applied our protocol on a subset of 20 HMEC samples, comprising 10 controls and 10 samples infected with adenovirus expressing activated H-Ras, thus extracting an alternative pathway signature for RAS oncogene. The classification task concerned with this data set is trivial since most classification algorithms can easily discriminate between the two classes. In this case, however, we are not interested in

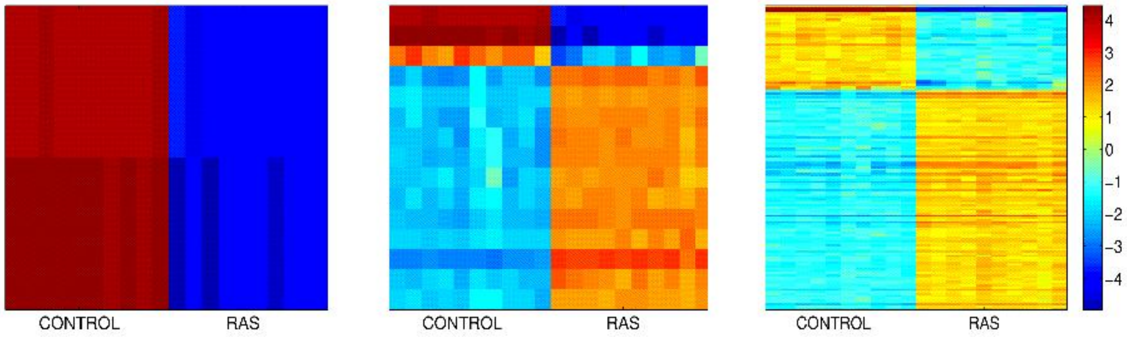


Figure 7.1: Image intensity display of the expression levels of the 2, 15 and 144 genes selected by ℓ^1-ℓ^2 regularization for $\mu = 0,\ 0.05,\ 0.5$ respectively. Expression levels are standardized to zero mean and unit variance across samples, displayed with genes as rows and samples as columns, and colour coded to indicate high (red) or low (blue) expression levels.

the classification performance on an independent test set, but in the selection of relevant gene lists and in their hierarchical structure. We thus apply our method to the RAS data and report the heat maps of the selected gene lists in Figure 7.1.

For $\mu = 0$, the method extracts a minimal set consisting of two probe sets of RAP1A (Figure 7.1, left), a gene belonging to the RAS oncogene family, whereas for increasing values of μ, the method selects perfectly nested larger sets of genes (probe sets) correlated or anti-correlated with the first two, but with lower fold change. In Figure 7.1, middle and right, we show the results obtained for $\mu = 0.05$ and $\mu = 0.5$ respectively (corresponding to 15 and 144 genes).

From the obtained results we can see that the method selects nested groups of genes which appear to be relevant to the RAS status, nicely sorted by their differential expressions. The two minimal probe sets are not part of the RAS signature defined in [18], whereas in the larger gene lists about 80% of the genes overlap with those found in [18] (12 out of 15 and 112 out of 144). Due to the higly controlled experimental setting, in this case only univariate dependance occur between input and output. It is thus unnecessary to apply the clustering technique described in Chapter 6, since all selecetd genes from a unique gene module.

7.1.2 Patient-tissue microarray data

We carried out experiments on benchmark patient-tissue data sets relative to three diseases: leukemia, lung cancer and prostate cancer.
Leukemia data set The first one is the well-known Golub data set [54] (available on line at `http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi`) which comprises expressions of 7129 probe sets for 72 patients (samples) divided in two classes according to the diagnosis of Acute Myeloid Leukemia (AML) or Acute Lymphoblastic Leukemia (ALL).
Lung Cancer data set The second data set [55] (which can be downloaded at `http://www.chestsurg.org`) consists of 181 lung cancer samples with 12533 probe sets. Each patients is

either affected by malignant pleural mesothelioma (MPM) or adenocarcinoma (ADCA).
Prostate Cancer data set The last data set we analyzed is the prostate cancer data set [100] which is available on line at `http://www-genome.wi.mit.edu/mpr/prostate`. It comprises 12533 probe sets for 102 samples, tumor or normal tissue.
In all cases, the vector Y is formed by labels $+1$ or -1 distinguishing the two classes.

Before running the two-stage procedure feature selection protocol described in Chapter 4, we verify the nesting property of the ℓ^1-ℓ^2 regularization solutions. To this aim, we applied to the entire data sets the damped thresholding algorithm without the acceleration, for increasing value of the ℓ^2 parameter μ, and report in Table 7.1 the number of selected genes, and the the percentage of these genes which are also present in the gene set selected with the next larger value of μ. As shown in Table 7.1, the hueristic for applying the acceleration seems to be well founded, since for the three data sets under consideration, the percentage of enclosure of each group in the group selected with the adjacent value of μ was always greater than 98%.

	Leukemia		Lung Cancer		Prostate Cancer	
μ	A	B	A	B	A	B
0	15	100%	17	100%	18	100%
$5 \cdot 10^{-4}$	33	98%	21	100%	31	100%
$1 \cdot 10^{-3}$	44	100%	26	100%	37	100%
$2 \cdot 10^{-3}$	70	99%	38	98%	59	100%
$5 \cdot 10^{-3}$	94		52		83	

Table 7.1: Nesting property of the proposed method. The first column contains the values of the parameter μ. For each of the three diseases the column A contains the number of selected genes, and the column B the percentage of these genes which are also present in the gene set selected with the next larger value of μ.

For the Leukemia data, we preserved the same data partition used in [54], where the 72 samples are divided in 38 training samples and 34 test samples; whereas for both lung and prostate cancer we split the data sets in two (almost) equal size subsets, taking care of maintaining the classes ratio. We then applied the two-stage feature selection procedure described in Chapter 4 to the 38, 91, and 51 training samples respectively, with the model selection protocol for intermediate sample size. In particular we performed leave-one-out cross-validation ($k = n_{train} = 38$), for Leukemia data and 10-fold cross-validation for lung and prostate cancer data.

A first indicator of the effectiveness of the proposed method is the stability of the various gene lists obtained in the training phase. In Figure 7.2, 7.3, and 7.4 we report the number of selected genes versus the selection frequency for different values of the parameter μ. By inspecting Figures 7.2, 7.3, and 7.4 one sees that the produced gene lists are remarkably stable. For increasing values of μ the number of genes appearing in all of the lists ranges from about 1/3 to about 1/2 of the average number of genes, while the number of genes appearing in at least 50% of the lists is very close to the average.

The accuracy of the method on the three data sets (which should remain the same for the different

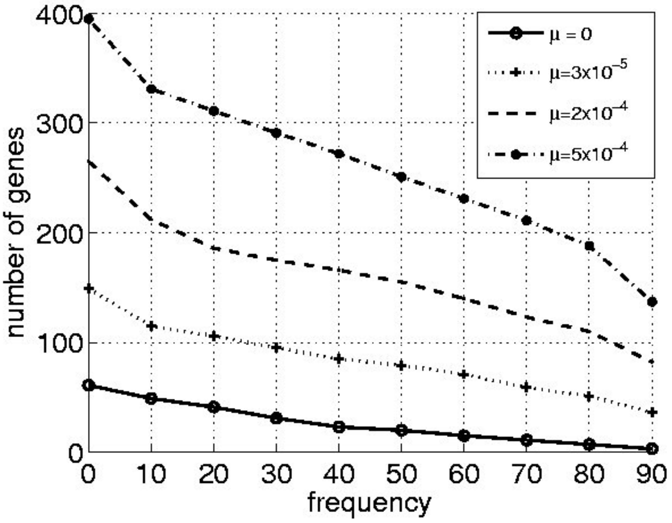


Figure 7.2: Cumulative number of selected genes versus selection frequency in LOO cross-validation for Leukemia data.

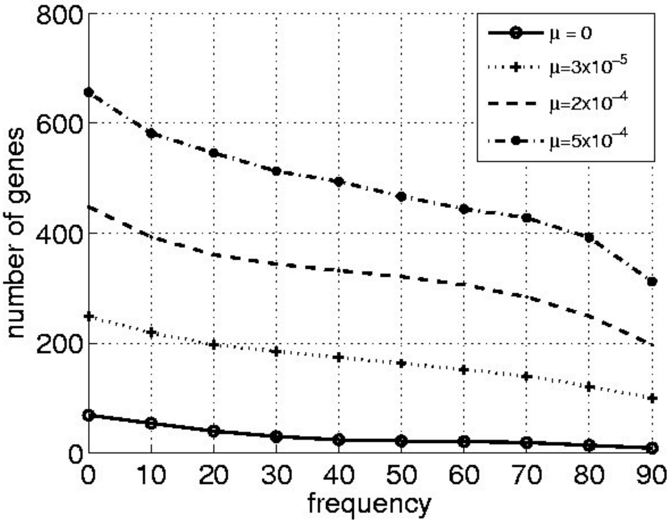


Figure 7.3: Cumulative number of selected genes versus selection frequency in 10-fold cross validation for Lung Cancer data

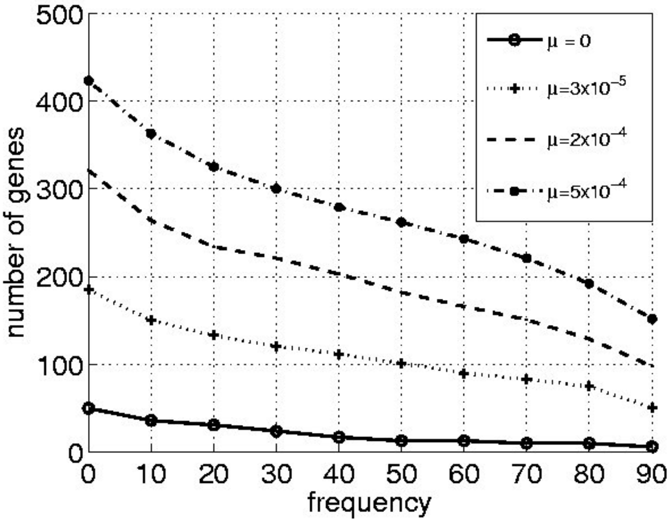


Figure 7.4: Cumulative number of selected genes versus selection frequency in 10-fold cross validation for Prostate Cancer data

values of μ) is illustrated in Table 7.2. By inspection we can see that for each disease and different values of μ (column A) the model cardinality from top to bottom increases (column B) while the prediction accuracy on the test set (column B) remains quite stable. For each disease in column B errors are reported for the two classes separately. The rightmost column C gives the percentage of samples which have to be rejected for both classes in order to reach 100% classification rate. The rejection region corresponding to $\mu = 0$ for the three diseases is depicted in Figure 7.5. The solid line gives the decision boundary, while the dashed lines mark the rejection region needed to reach the perfect score. No rejection region is needed for the Leukemia study (Figure 7.5, left), a one-sided rejection region for the lung cancer study (Figure 7.5, middle) and a wider two-sided rejection region for the prostate cancer case (Figure 7.5, right).

Leukemia ($n_{test} = 34$)				Lung Cancer ($n_{test} = 90$)				Prostate Cancer ($n_{test} = 51$)			
μ	A	B	C	μ	A	B	C	μ	A	B	C
0	28	(0,0)	(0%, 0%)	0	34	(1,0)	(0%,3%)	0	21	(2,1)	(13%,48%)
$6{\cdot}10^{-6}$	32	(0,1)	(5%, 0%)	$2{\cdot}10^{-7}$	37	(1,0)	(0%,3%)	10^{-6}	26	(2,0)	(0%,54%)
$1{\cdot}10^{-5}$	34	(0,2)	(9%, 0%)	$3{\cdot}10^{-6}$	51	(1,0)	(0%,1%)	$5{\cdot}10^{-6}$	29	(2,0)	(0%,46%)
$3{\cdot}10^{-5}$	40	(0,2)	(18%, 0%)	$9{\cdot}10^{-6}$	78	(1,0)	(0%,1%)	$3{\cdot}10^{-5}$	45	(3,1)	(4%,46%)
$5{\cdot}10^{-5}$	50	(0,3)	(39%, 0%)	10^{-5}	108	(1,0)	(0%,1%)	$4{\cdot}10^{-5}$	58	(2,0)	(0%,46%)
10^{-4}	71	(1,1)	(20%,14%)	$3{\cdot}10^{-5}$	152	(1,0)	(0%,1%)	$6{\cdot}10^{-5}$	72	(2,0)	(0%,46%)
$2{\cdot}10^{-4}$	108	(1,2)	(14%, 8%)	$5{\cdot}10^{-5}$	174	(1,0)	(0%,1%)	10^{-4}	108	(2,0)	(0%,50%)
$4{\cdot}10^{-4}$	135	(1,2)	(14%,15%)	$7{\cdot}10^{-5}$	211	(1,0)	(0%,1%)	$4{\cdot}10^{-4}$	195	(2,0)	(0%,54%)

Table 7.2: Prediction accuracy of the proposed method on patients-tissue microarray data. For each of the three diseases the first column contains the values of the parameter μ, the column A the number of selected genes, the column B the number of misclassified samples for the two original classes respectively, and the column C the percentage of samples to be rejected in each predicted class in order to obtain 100% classification rate. The two classes are (ALL, AML) for Leukemia, (MPM,ADCA) for Lung Cancer, and (normal, tumor) for Prostate Cancer.

An improvement in prediction accuracy is not the aim of the proposed method. However, it

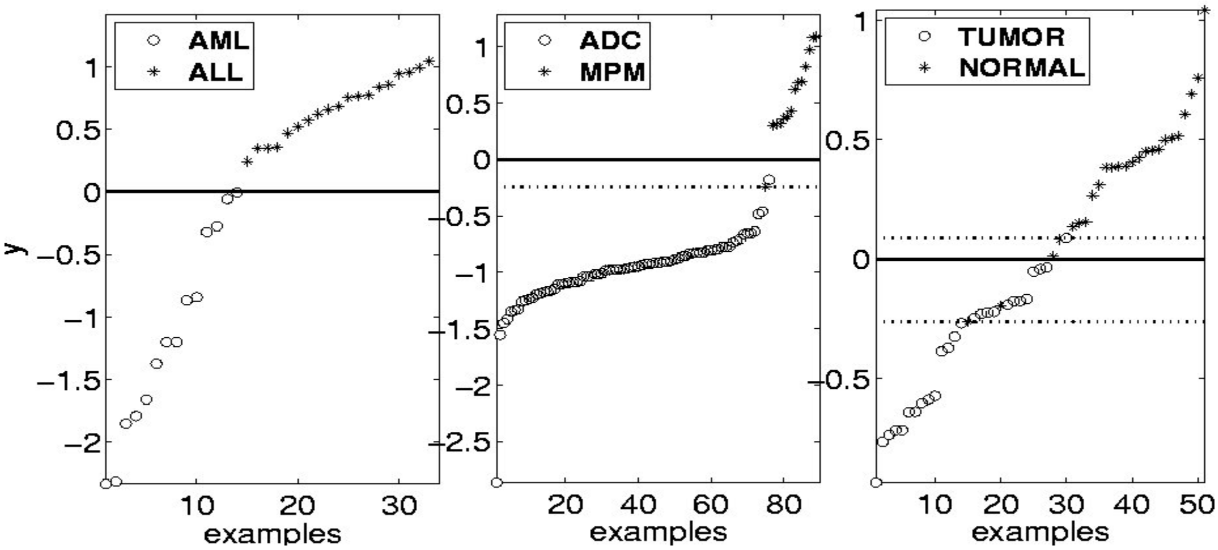


Figure 7.5: Rejection region for $\mu = 0$. In Leukemia (left) the score is perfect and the region is degenerate, in Lung Cancer (middle) is one-sided and delimited by the dashed line, in Prostate Cancer (right) is two sided.

is interesting to notice that the proposed method reaches performances which are at least as good as and often better than those reported in the original studies. In the leukemia original paper [54], a 50-genes classifier is built which scored 100% on the test set, though only 29 of the 34 test samples corresponded to strong prediction (i.e. prediction with a high confidence level). The prediction accuracy of our method ranges from 91% to 100%. As for the lung cancer data analysis in [55], different classifiers were reported with prediction accuracy ranging from 91% to 99%, to be compared with the 99% achieved with our algorithm. In the end, for the prostate cancer data set, in [100] – after gene ranking with variation of a signal-to-noise metric – a k-NN algorithm obtained a prediction accuracy ranging from 82.9% to 95.7% depending on the number of genes used (4 or 6); with our method the accuracy ranges from 92% to 96%.

Where available (leukemia and lung cancer), we have compared the gene lists we obtained with the lists produced by other methods. The results show partial superposition (depending on μ) as well as important differences. The difference between our results and the ones reported in the original papers is not surprising given the multivariate flavor of our selection procedure. Ultimately, only biological validation can assess the actual relevance of the gene lists obtained by different methods.

In unsupervised cluster analysis, despite a large number of heuristically motivated methods, there are no theoretically founded approaches capable to assess the goodness of the clusters. therefore we can accomplish such validation task only in a indirect way, by means of the two visualization tools proposed in Chapter 6 which qualitatively confirm the methodology effectiveness in the three data sets. In fact the well defined color patches in Figure 7.6, 7.7, 7.8 (top) and the clusters plotted in Figure 7.6, 7.7, 7.8 (bottom) clearly indicate that the proposed technique does detect a strong correlation pattern.

μ	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}
0	1	1	1	1	1	1	1	1	1	1
$4 \cdot 10^3$	1	2	2	1	2	2	1	1	1	1
$3 \cdot 10^2$	1	3	4	1	2	2	3	2	1	1
$2 \cdot 10^1$	2	5	9	1	3	6	4	6	2	1
1.7	3	8	25	1	7	23	11	11	23	2
13	10	28	101	4	46	77	22	19	71	10
100	28	36	239	10	91	151	38	25	131	26

Table 7.3: Number of genes in the top 10 blocks, $B_1, \ldots, B_{10}$, at increasing values of μ for Leukemia data.

μ	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}
0	1	1	1	1	1	1	1	1	1	1
$4 \cdot 10^3$	1	2	1	1	1	3	1	1	1	1
$3 \cdot 10^2$	1	11	2	2	2	4	2	1	2	1
$2 \cdot 10^1$	1	27	4	4	3	6	3	8	5	7
1.7	9	61	15	6	6	13	10	19	29	11
13	30	108	32	15	12	30	21	34	73	33
100	59	140	68	24	20	57	33	78	103	53

Table 7.4: Number of genes in the top 10 blocks, $B_1, \ldots, B_{10}$, at increasing values of μ for Lung data.

μ	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}
0	1	1	1	1	1	1	1	1	1	1
$4 \cdot 10^3$	2	2	1	1	2	1	1	1	1	1
$3 \cdot 10^2$	2	4	2	1	2	2	1	3	3	1
$2 \cdot 10^1$	7	9	5	2	4	2	1	8	5	3
1.7	12	28	21	3	8	4	6	17	5	7
13	27	40	48	3	10	7	6	21	8	9
100	31	49	60	3	11	8	6	22	10	11

Table 7.5: Number of genes in the top 10 blocks, $B_1, \ldots, B_{10}$, at increasing values of μ for Prostate data.

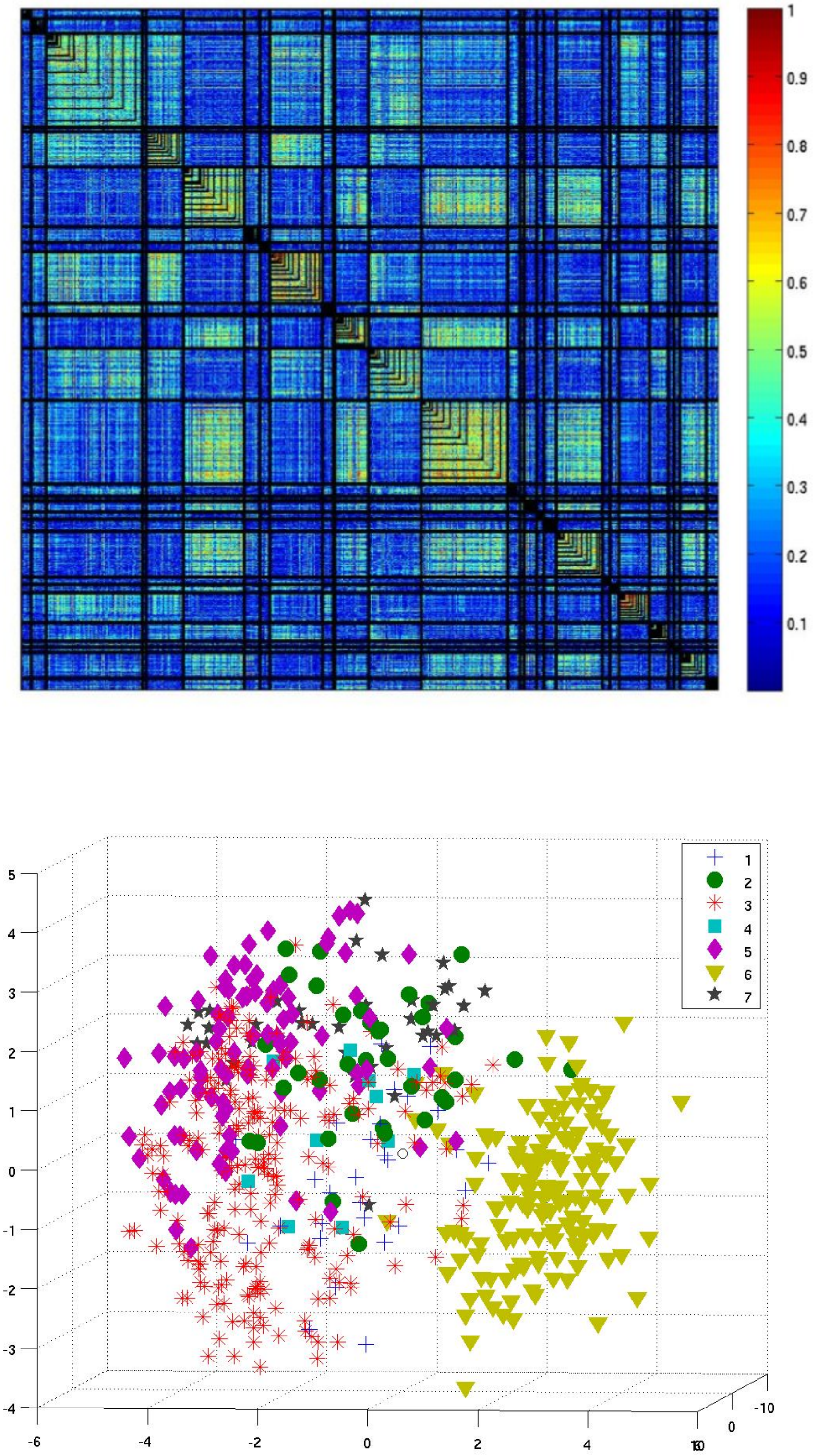


Figure 7.6: Correlation and clustering in leukemia data.

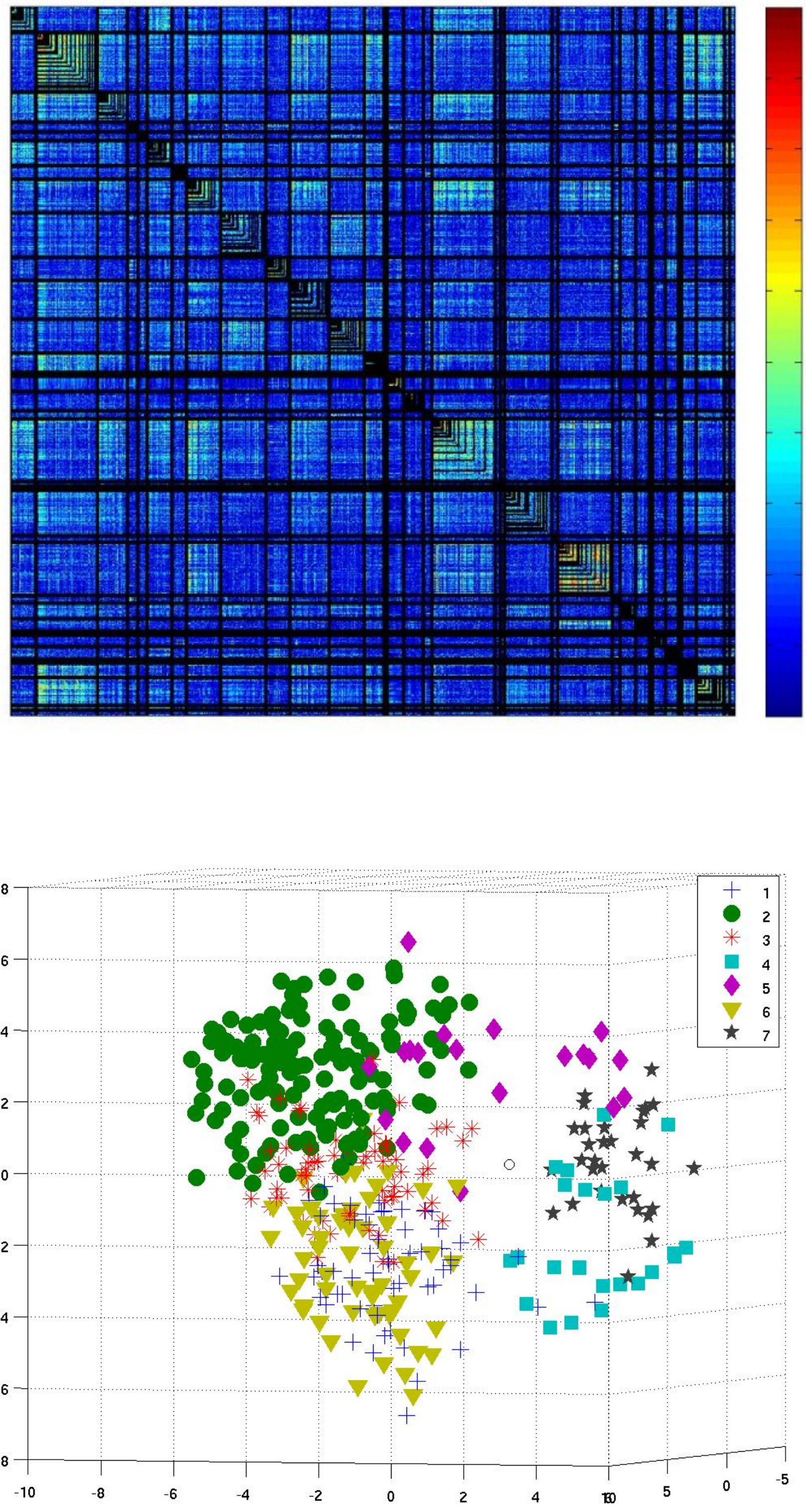


Figure 7.7: Correlation and clustering in lung cancer data.

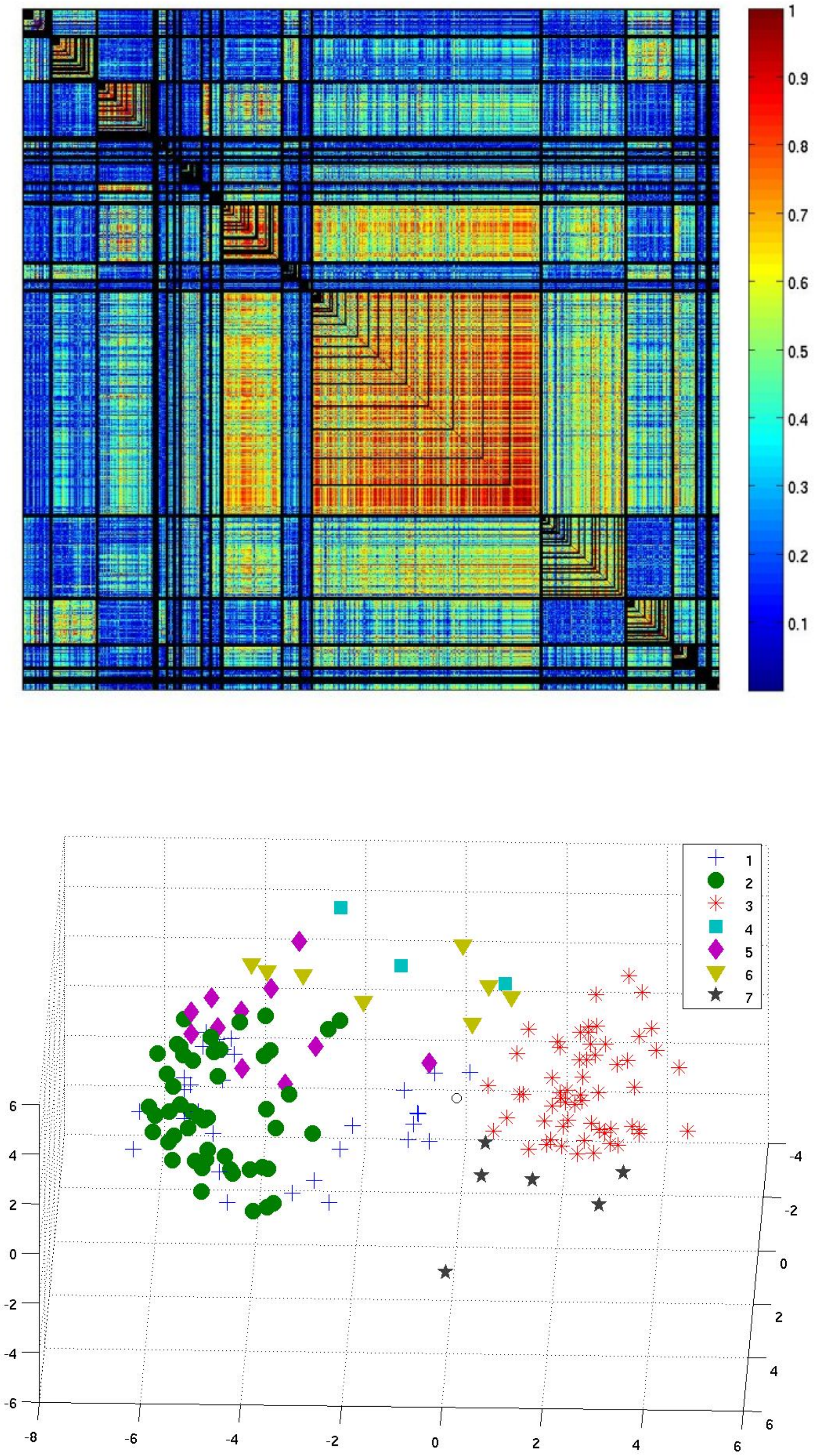


Figure 7.8: Correlation and clustering in prostate cancer data.

Where available (leukemia and lung cancer), we have compared the gene lists we obtained with the lists produced by other methods. The results show partial superposition (depending on μ) as well as important differences. Such differences are not surprising given the multivariate flavor of our feature selection procedure. As for the the layered and ordered clusters provided by our technique, their relevance cannot be directly assessed. In fact, in unsupervised cluster analysis, despite a large number of heuristically motivated methods, there are no theoretically founded approaches capable to assess the goodness of the clusters. Neverthless, the two proposed visualization tools qualitatively confirm the methodology effectiveness. Let us remark that only biological validation can assess the actual relevance of both gene lists and their structure.

7.2 Hypoxia signature in neuroblastoma cell llines

In this section we present an ongoing work in collaboration with the molecular biology laboratory of the children hospital Giannina Gaslini in Genova. We are interested in the Biology of neuroblastoma, the most common pediatric solid tumor, and in its conditioning by the tumor microenvironment. Hypoxia is a low oxygen situation that occurs in the developing of tumor mass and that is associated with poor prognosis and resistance to chemo- and radio-therapy. Hypoxia has a profound effect on the biology of the cell. The challenge is to devise methods suitable for deriving biologically relevant signatures form in vitro controlled systems for prognostic/diagnostic applications (in vivo applications to the patients). We applied a combination of supervised and unsupervised learning techniques to define the hypoxia signature of neuroblastoma cell lines.

7.2.1 Biological background

Neuroblastoma is the most common paediatric solid tumor, usually deriving from immature or precursor cells of the ganglionic lineage of the sympathetic nervous system (SNS) [32, 105]. It is the most common tumor diagnosed during infancy and shows notable heterogeneity, with regard to both histology and clinical behavior [77], ranging from rapid progression associated with metastatic spread and poor clinical outcome to occasional, spontaneous, or therapy-induced regression or differentiation into benign ganglioneuroma [112]. The local oxygenation status is one important parameter determining the tumor phenotype at a regional level, and in numerous reports, oxygen shortage low O2 tension (hypoxia) has been shown to have profound effects on tumor cell phenotype [58]. Rapidly expanding neuroblastoma tumors present areas of hypoxia and metastasize to hypoxic sites, such as bones and bone marrow [80]. Hypoxia is a common denominator of many pathologic processes and a critical determinant of tumor cell growth, susceptibility to apoptosis, and resistance to radio- and chemotherapy [58]. Hypoxia has a profound impact on neuroblastoma aggressive behavior. Increased production of the angiogenic mediator, vascular endothelial growth factor (VEGF), and of the Id2 protein, an inhibitor of the antiproliferative function of the retinoblastoma (Rb) tumor suppressor gene [72], was observed in hypoxic neuroblastoma cells both in vitro and in vivo [65] Furthermore, hypoxia down-regulates several neuronal/neuroendocrine marker genes in neuroblastoma cells and, conversely, up-regulates genes expressed in neural crest sympathetic progenitors, leading to neuroblastoma cell de-differentiation and acquisition of an immature and more aggressive neural-crest-like phenotype [65]. Although hypoxia has been recognized as an important determinant of clinical outcomes in human cancers [98], it has been difficult to perform comprehensive and quantitative analyses to define tumor phenotypes based on hypoxia responses and to explore the relationship between tumor hypoxia and genetic changes or clinical parameters in human cancers.

7.2.2 Data Collection

Nine human neuroblastoma cell lines GI-LI-N, ACN, GI-ME-N, IMR-32, SHEP-2, LAN-1, SK-N-BE(2c), SK-N-F1, and SK-N-SH were cultured in RPMI 16140 (Euroclone Ltd., Celbio, Milan, Italy), supplemented with 10% heat-inactivated fetal bovine serum (Sigma, Milan Italy), 2 mmol/L L-glutamine, 10 mM Hepes, 100 units/mL penicillin, and 100 g/mL streptomycin (Euroclone Ltd), at 37C in a humidified incubator containing 20% O2, 5% CO2, and 75% N2. Hypoxic conditions (1% O2) were achieved by culturing the cells in an anaerobic workstation incubator (BUG BOX, Jouan, ALC International S.r.l., Cologno Monzese, Milano, Italy) flushed with a gas mixture containing 1% O2, 5% CO2, and balanced N2 at 37C in a humidified atmosphere. Oxygen tension in the medium was measured with a portable, trace oxygen analyzer (Oxi 315i/set, WTW; VWR International, Milano, Italy).
Total RNA from neuroblastoma cell lines in normoxic and hypoxic conditions was reverse transcribed into cDNA and biotin labeled. Biotin-labeled cRNA was cleaned up with the Qiagen RNeasy Mini kit and ethanol precipitation, checked for quality with Agilent Bioanalyzer 2100, and fragmented by incubation at 94C for 35 min in 40 mmol/L Tris-acetate (pH 8.1), 100 mmol/L potassium acetate, and 30 mmol/L magnesium acetate. Fragmented cRNA was used for hybridization to Affymetrix HG-U133 Plus 2.0 arrays (Affymetrix, Santa Clara, CA). GeneChips were scanned using an Affymetrix GeneChip Scanner 3000. All microarrays were examined for surface defects, grid placement, background intensity, housekeeping gene expression, and a 3:5 ratio of probe sets from genes of various lengths. Gene expressions were then extracted from CEL files and normalized using the Robust Multichip Average (RMA) method.

7.2.3 Data analysis

The problem was cast as a supervised variable selection problem and the accelerated $\ell^1 - \ell^2$ regularization procedure described in the previous chapters was applied. A classification rule is built able to discriminate the cell lines depending on their hypoxic status. Note that in this approach a gene is considered to be relevant if it contributes in building a multivariate discriminative model for the hypoxic status. This can be compared with the t-test approach where a gene is defined to be relevant if the expression means for the two statuses are significantly different according to the t-statistics. In order to have a benchmark to compare our results with we tested the hypothesis of equal distribution of the gene in the two different status by means of t-statistic. Since we want to have a significant test for many genes we correct for multiple hypothesis testing with Benjamini and Hochberg method for controlling the False Discovery Rate [28]. However, no gene passed the test.

Due to the low number of samples available we applied the double loop cross validation framework described in Chapter 4. In this condition we do not have a unique set of features, since we extract a possibly different solution at each iteration of the outer loop. We therefore have to merge these partially overlapping gene sets in order to derive a unique list. Toward this end we sorted the union of all gene lists according to the single gene selection frequency. Since we run our selection protocol with two different values of the correlation parameter ϵ. With $\epsilon = 1$ we extract the minimal list, whereas with $\epsilon = 100$ we obtain the correlation aware list. The effect of changing the correlation coefficient is showed in the Figure 7.9. As can be seen from the

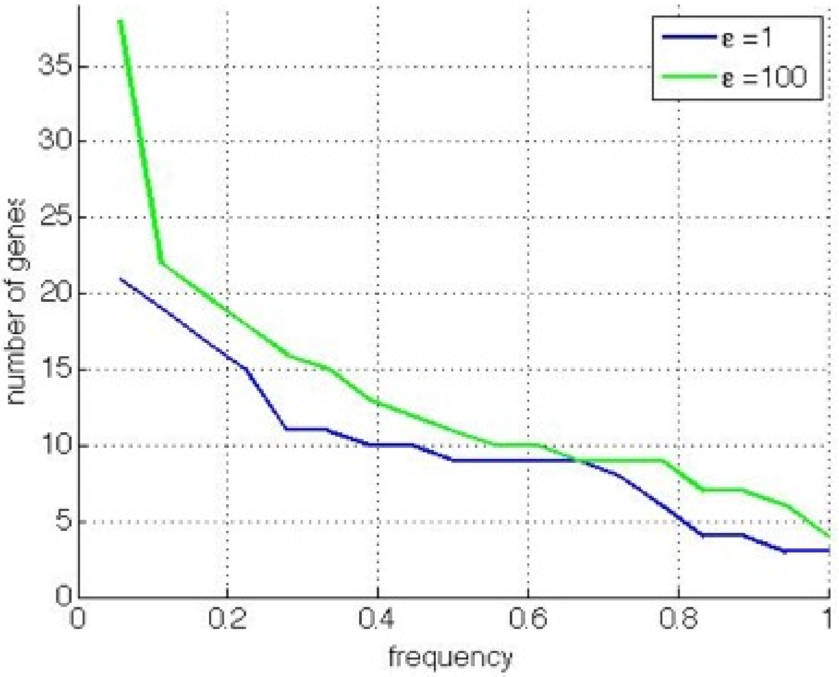


Figure 7.9: Relative frequency for $\epsilon = 1$ and $\epsilon = 100$.

Figure 7.9 when the relative frequency is above roughly 70% (13/18) we select 9 genes for the correlation ware list and 8 for the minimal list. The leave one out error is 3 out of 18 ($\epsilon = 1, 100$s).

We further reported in Table 7.6 the gene selected in the correlation aware lists. Genes are sorted according to their selection frequency. We discarded genes with a frequency below 70%. The table includes the affy ID, percentage frequency of selection in leave-one-out cross-validation, gene symbol, gene description, and GenBank ID.

We further visualize the correlation aware signature (ref. Table 7.6) in several ways. In Figure 7.10, we provide a 3-dimensional visualization of the data set restricted to the 9 probesets. In order to obtain a 3D representation the data submatrix is projected on its 3 principal components, i.e. the components of maximum variance. In Figure 7.11 we provide a univariate representation of the log-scale expression of each of the 9 probesets. It is worthing noting that while in Figure 7.10 the two classes are well separated in the 3-dimensional space, by projecting on the single probeset, Figure 7.11, the two classes do not cluster as well.

affy ID	frequency	Symbol	Description	GenBank
201848_s_at	100	BNIP3	BCL2/adenovirus E1B 19kDa interacting protein 3	U15174
202887_s_at	100	DDIT4	DNA-damage-inducible transcript 4	NM_019058
226452_at	100	PDK1	pyruvate dehydrogenase kinase; isoenzyme	1 AU146532
236180_at	100		Transcribed locus; weakly similar to NP_062553.1 hypothetical protein FLJ11267 [Homo sapiens]	W57613
223193_x_at	94	E2IG5	growth and transformation-dependent protein	AF201944
225342_at	94	AK3L1	adenylate kinase 3-like 1	AK026966
224345_x_at	89	E2IG5	growth and transformation-dependent protein	AF107495
202022_at	78	ALDOC	aldolase C; fructose-bisphosphate	NM_005165
210512_s_at	78	VEGF	vascular endothelial growth factor	AF022375

Table 7.6: Hypoxia Signature associated to neuroblastoma cell lines for $\epsilon = 100$.

By an initial a posteriori biologcal analysis it's worth mentioning that all the probe sets in the signature are known for being modulated by hypoxia. This means that the experiment allowed to identify a subset of genes modulated by hypoxia in neuroblastoma. In order to compare our method with a benchmark technique we tested the hypothesis of equal distribution of the gene in the two different status by means of t-statistic. Since we want to have a significant test for many genes we correct for multiple hypothesis testing with Benjamini and Hochberg method for controlling the False Discovery Rate (FDR). It is interesting to mention when analyzing all 54000 probe set present on the chip, due to the FDR correction, none of them has a significant p-value. This therefore is a clear example where a supervised learning technique, based on $\ell^1 - \ell^2$ regularization principle combined with cross validation framework allowed the definition of an hypoxia signature with specific characteristics of robustness and reproducibility superior to those generated by canonical statistical tests.

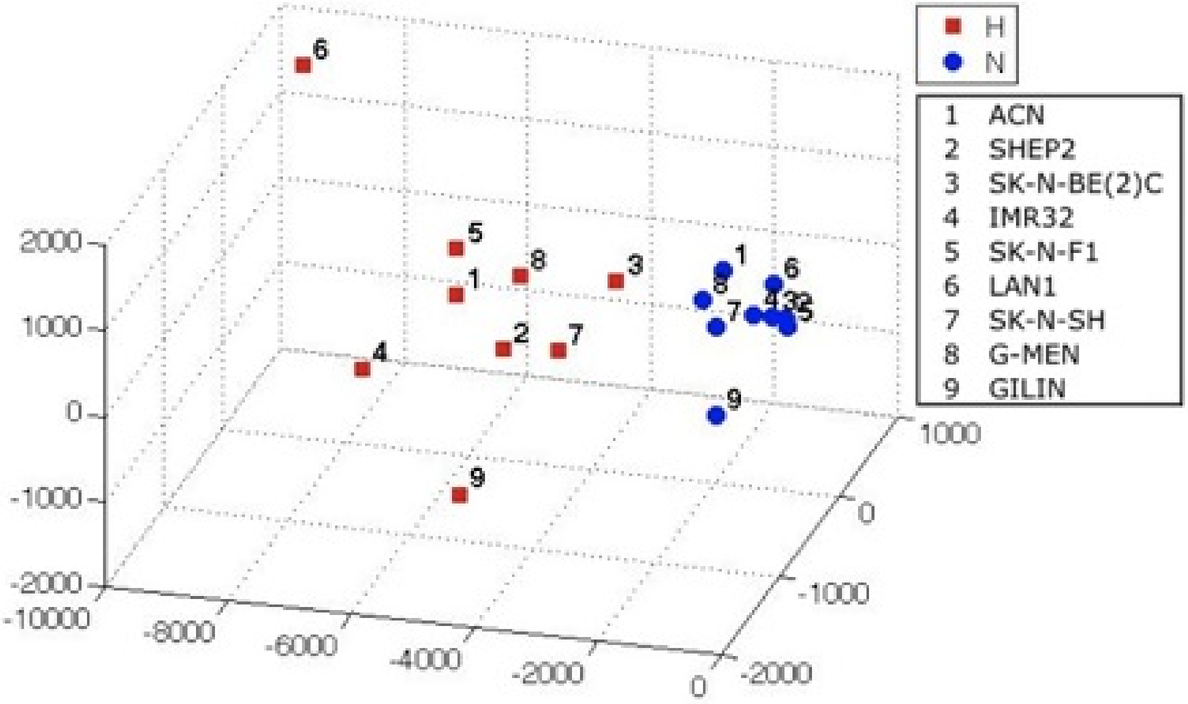


Figure 7.10: 3D representation of the cell lines based on the correlation aware gene signature. The red square represents a cell line in hypoxic status, whereas the blue circle indicates normoxia.

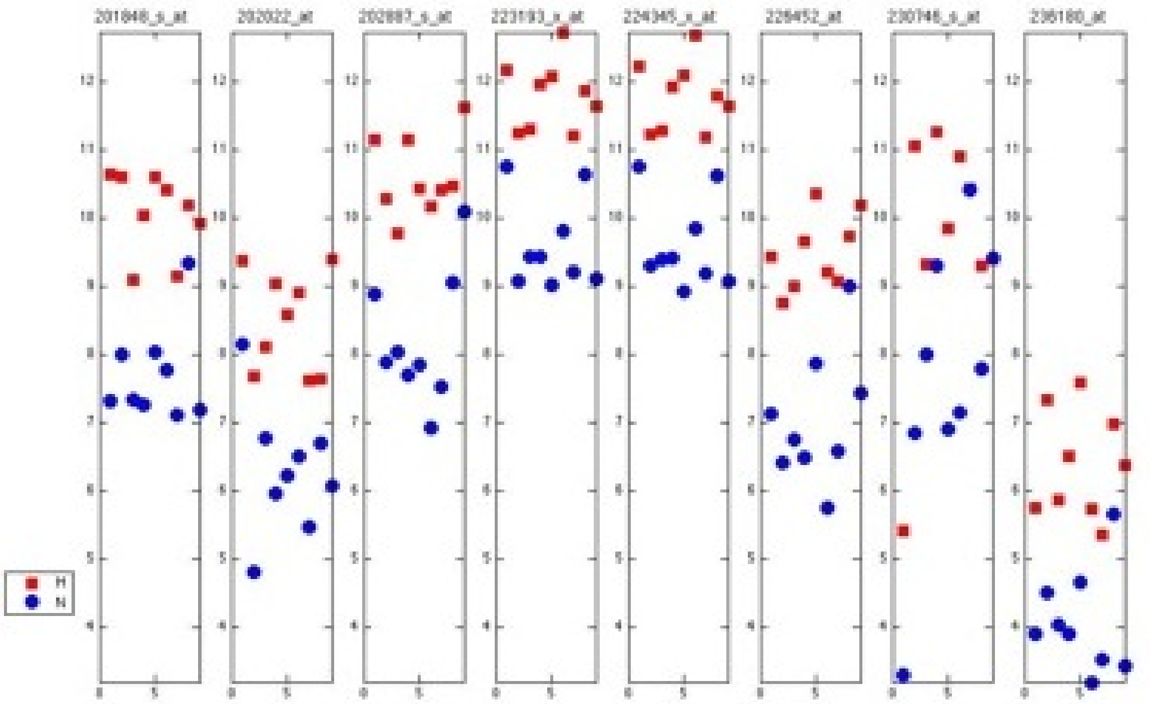


Figure 7.11: Univariate representation of the cell lines based on the correlation aware gene signature. For each selected probeset the log-scale expression is reported. The red square represents a cell line in hypoxic status, whereas the blue circle indicates normoxia.

Chapter 8

Conclusions

In this thesis we have dealt with the problem of extracting subsets of relevant features from high-dimensional data, with particular attention to the application to microarray gene expression data analysis. First we have theoretically motivated the need for feature selection, based on regularization properties of dimensionality reduction. Indeed we have shown that an accurate reduction of the input data dimensionality has a positive effect on the prediction power of the projected data, thus discrediting the misleading intuition that by adding more dimensions, and therefore more information from the data distribution, results should improve.

Following the theoretical motivations for performing feature selection, we presented our proposal of a robust statistical analysis protocol for identifying nested subsets of relevant variables in the learning from examples framework. Being formulated as a convex optimization problem, our learning method has a sound mathematical foundation and, moreover, the models can be efficiently computed through simple and easy-to-implement algorithms. It is particularly appealing for the study of gene expression data since it relies on a truly multivariate analysis and, in contrast to the usual gene-by-gene analysis, does not only rank genes on the basis of their differential expression on the samples, and takes into account the correlation patterns arising from the organization of genes in cooperating networks. The two main features of the proposed method are that it provides nested lists of genes and that the genes additionally included in the longer lists are correlated with the genes of the shorter ones. Both these properties can be very helpful when analyzing high-throughput data and might shed light on the biological mechanisms under study. However, since the obtained models are asymptotically equivalent in terms of prediction accuracy, the choice of which list is the most appropriate is left to the molecular biologist and ultimately depends on the underlying question and the available prior knowledge.

The prediction power and (where possible) the accuracy of the model associated with the gene signatures identified by our procedure are confirmed by experiments both on synthetic and real data. In particular the proposed model allowed the definition of an hypoxia signature associated to in vitro neuroblastoma cell lines, with specific characteristics of robustness and reproducibility superior to those generated by canonical statistical tests. Nevertheless, despite these appealing results, the biological interpretability of the selected gene lists is often a major problem. To this aim we proposed an ad hoc agglomerative clustering technique able to refine such a nested output by explicitly identifying modules of correlated genes and rank them according to their discriminative power. Furthermore, in order to enhance and easily interpret the obtained structure we propose two ad hoc visualization tools. In this way we can extract and visualize a more structured genes signature, which captures and make evident the correlation patterns in the data. This provides a richer model, that can be used to gain a better understanding of the genes function, possibly leading to new biological hypotheses.

Concluding, in the context of variable selection, in particular of sparse regularization, an appealing open problem in machine learning is concerned with the development of efficient and theoretically founded techniques capable of inserting different kinds of prior knowledge into the regularization procedure; important examples are the problem of performing feature selection though modeling nonlinear dependences between the inputs and the outputs, and the problem of taking into account some knowledge of the regulation networks among genes. Toward this end we have developed a general unifying framework for solving a large class of sparse regularization problems. Thanks to its flexibility, the proposed framework for sparse regularization is highly customizable and can thus be of great help, since the underpinning theorems hold under very broad assumptions. Indeed, one can focus on building an appropriate penalty encoding his subjective knowledge of the underlying phenomenon. Then, once a suitable (possibly non differentiable) penalty term will be found, if the one-homogeneity hypothesis is satisfied, the framework guarantees a ready-to-use iterative algorithm for solving the corresponding minimization problem.

Bibliography

[1] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc Natl Acad Sci U S A*, 96(12):6745–6750, June 1999.

[2] E. Amaldi and V. Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theor. Comput. Sci.*, 209(1-2):237–260, 1998.

[3] C. Ambroise and G.J. McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Natl. Acad. Sci. USA.*, 99(10):6562–6566, 2002.

[4] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, 2005.

[5] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In Bernhard Schlkopf, John Platt, and Thomas Hoffman, editors, *NIPS*, pages 41–48. MIT Press, 2006.

[6] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

[7] S. Avidan. Spatialboost: Adding spatial reasoning to adaboost. In *ECCV06*, pages IV: 386–396, 2006.

[8] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In Carla E. Brodley, editor, *ICML*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004.

[9] A. Barla, S. Mosci, L. Rosasco, and A. Verri. A method for robust variable selection with significance assessment. In *Proc. of ESANN, European Symposium on Artificial Neural Networks*, 2008.

[10] P.L. Bartlett, M.J. Jordan, and J.D. McAuliffe. Convexity, classification, and risk bounds. Technical Report 638, Department of Statistics, U.C. Berkeley, 2003.

[11] F. Bauer and S. Pereverzev. Regularization without preliminary knowledge of smoothness and error behaviour. *European J. Appl. Math.*, 16(3):303–317, 2005.

[12] F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *Journal of complexity*, 23:52–57, 2006.

[13] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56:209–239, 2004.

[14] M. Belkin, P. Niyogi, and V. Sindhwani. On Manifold Regularization. In *AISTAT*, 2005.

[15] R. Bellman. *Adaptive Control Processes: A Guided Tour.* Princeton University Press, 1961.

[16] S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Proc. of the Sixteenth Annual Conference on Learning Theory COLT 2003*, 2003.

[17] M. Bertero and P. Boccacci. *Introduction to Inverse Problems in Imaging.* Institute of Physics Publishing, 1998.

[18] A.H. Bild, G. Yao, J.T. Chang, Q. Wang, A. Potti, D. Chasse, M. Joshi, D. Harpole, J.M. Lancaster, A. Berchuck, J.A. Olson, J.R. Marks, H.K. Dressman, M. West, and J.R. Nevins. Oncogenic pathway signatures in human cancers as a guide to targeted therapies. *Nature*, November 2005.

[19] G. Blanchard, P. Massart, R. Vert, and L. Zwald. Kernel projection machine: a new tool for pattern recognition. In *NIPS 2004*, pages 1649–1656, 2004.

[20] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artif. Intell.*, 97(1-2):245–271, 1997.

[21] L. Breiman, C. J. Stone, R. A. Olshen, and J. H. Friedman. Classification and regression trees. In *Wadsworth and Brooks*, 1984.

[22] F. Bunea, A.B. Tsybakov, and M. H. Wegkamp. Aggregation and sparsity via l1 penalized least squares. In Gbor Lugosi and Hans-Ulrich Simon, editors, *COLT*, volume 4005 of *Lecture Notes in Computer Science*, pages 379–391. Springer, 2006.

[23] E. Candes and T. Tao. The dantzig selector: statistical estimation when p is much larger than n, 2005.

[24] E. Candes and T. Tao. Discussion: The dantzig selector: statistical estimation when p is much larger than n. *Annals od Statistics*, 35(6):2365–2369, 2007.

[25] A. Caponnetto and E. De Vito. Optimal rates for regularized least-squares algorithm. *Found. Comput. Math. In Press, DOI 10.1007/s10208-006-0196-8. Online August 2006*, 2006.

[26] A. Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vis.*, 20(1-2):89–97, 2004.

[27] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[28] R.R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis: Special issue on Diffusion Maps and Wavelets*, 21:5–30, 2006.

[29] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57:1413–1457, 2004.

[30] C. De Mol, E. De Vito, and L. Rosasco. Elastic-net regularization in learning theory, 2008.

[31] C. De Mol, Mosci, M. S. Traskine, and A. Verri. A regularized method for selecting nested groups of relevant genes from microarray data. *Journal of Computational Biology*, XX, 2008.

[32] K. De Preter, J. Vandesompele, P. Heimann, N. Yigit, S. Beckman, A. Schramm, A. Eggert, R.L. Stallings, Y. Benoit, M. Renard, A. De Paepe, G. Laureys, S. Pahlman, and F. Speleman. Human fetal neuroblast and neuroblastoma transcriptome analysis confirms neuroblast origin and highlights neuroblastoma candidate genes. *Genome Biology*, 7, 2006.

[33] E. De Vito, L. Rosasco, A. Caponnetto, U. De Giovannini, and F. Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6:883–904, 2005.

[34] A. Destrero, S. Mosci, C. De Mol, A. Verri, and F. Odone. Feature selection for high dimensional data. *Computational Management Science*, to appear.

[35] David L. Donoho. For most large underdetermined systems of equations, the minimal l_1-norm near-solution approximates the sparsest near-solution. *Comm. Pure Appl. Math.*, 59(7):907–934, 2006.

[36] A. L. Dontchev and T. Zolezzi. *Well-posed optimization problems*, volume 1543 of *Lecture Notes in Mathematics*. Springer-Verlag, 1993.

[37] B. Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382):316–331, 1983.

[38] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

[39] B. Efron, R. Tibshirani, J.D. Storey, and V. Tusher. Empirical bayes analysis of a microarray experiment. *J. Amer. Stat. Assoc.*, 96:1151–1160, 2001.

[40] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap.* Chapman & Hall, New York, 1993.

[41] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, 95(25):14863–14868, December 1998.

[42] David Eisenberg, Edward M. Marcotte, Ioannis Xenarios, and Todd O. Yeates. Protein function in the post-genomic era. *Nature*, 405:823–826, May 2008.

[43] I. Ekeland and R. Temam. *Convex analysis and variational problems.* North-Holland Publishing Co., Amsterdam, 1976.

[44] H. W. Engl, M. Hanke, and A. Neubauer. Regularization of inverse problems. *Mathematics and its Applications*, 375, 1996.

[45] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[46] G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. of Machine Learning Research*, 3:1289–1306, 2003.

[47] M. Fornasier, I. Daubechies, and I. Loris. Accelerated projected gradient methods for linear inverse problems with sparsity constraints. *J. Fourier Anal. Appl.*, 2008.

[48] M. Fornasier and H. Rauhut. Recovery algorithms for vector-valued data with joint sparsity constraints. *SIAM J. Numer. Anal.*, 46(2):577–613, 2008.

[49] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and SYstem Sciences*, 55(1):119–139, 1997.

[50] Yoav Freund. Boosting a weak learning algorithm by majority. *Inf. Comput.*, 121(2):256–285, 1995.

[51] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting, 1998.

[52] C. Furlanello, M. Serafini, S. Merler, and G. Jurman. Entropy-based gene ranking without selection bias for the predictive classification of microarray data. *BMC Bioinformatics*, 4:54, 2003.

[53] D. Ghosh and A. M. Chinnaiyan. Classification and selection of biomarkers in genomic data using lasso. *J. Biomed. Biotechnol.*, 2:147–154, 2005.

[54] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286, 1999.

[55] R. Gordon, G. J.and Jensen, L.-L. Hsiao, S. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. Sugarbaker, and Raphael Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 62:4963–4967, 2002.

[56] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.

[57] C.A. Harrington, C. Rosenow, and J. Retief. Monitoring gene expression using dna microarrays. *Curr. Opin. Microbiol*, 3:285–291, 2000.

[58] A.L. Harris. Hypoxiaa key regulatory factor in tumor growth. *Nat.Rev. Cancer*, 2, 2002.

[59] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer-Verlag, 2001.

[60] Trevor Hastie, Robert Tibshirani, Michael Eisen, Ash Alizadeh, Ronald Levy, Louis Staudt, Wing Chan, David Botstein, and Patrick Brown. 'gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, 1(2), 2000.

[61] J.S.U. Hjorth. *Computer Intensive Statistical Methods: Validation, Model Selection and Bootstrap.* Chapman & Hall, 1994.

[62] A. E. Hoerl and R. Kennard. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

[63] A. Hyvarinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13:411–430, 2000.

[64] A.K. Jain and W.G. Waller. On the optimal number of features in the classification of multivariate gaussian data. *Pattern Recognition*, pages 365–374, 1978.

[65] A. Jogi, I. Ora, and H. et al. Nilsson. Hypoxia alters gene expression in human neuroblastoma cells toward an immature and neural crest-like phenotype. *Proc Natl Acad Sci USA*, 99:7021 7026, 2002.

[66] G. H. John, R. Kohavi, and P. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, New Brunswick, NJ, 1994. San Francisco: Morgan Kaufmann.

[67] G. Kerkyacharian and D. Picard. Thresholding in learning theory. *Constructive Approximation*, 26(2):173–203, 2007.

[68] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145, 1995.

[69] R. Kohavi and G. John. Wrappers for feature selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[70] G. R. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, November 2004.

[71] S. Lang. *Real analysis.* Addison-Wesley Publishing Company Advanced Book Program, Reading, MA, second edition, 1983.

[72] A. Lasorella, M. Noseda, M. Beyna, Y. Yokota, and A. Iavarone. Id2 is a retinoblastoma protein target and mediates signalling by myc oncoproteins. *Nature*, 407:592 598, 2000.

[73] C. Leng, Y. Lin, and G. Wahba. A note on the lasso and related procedures in model selection. *Statistica Sinica*, 16:1273–1284, 2006.

[74] Stan Z. Li and ZhenQiu Zhang. FloatBoost learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9), 2004.

[75] Y. Li, C. Campbell, and M. Tipping. Bayesian automatic relevance determination algorithms for classifying gene expression data. *Bioinformatics*, 18:1332–1339, 2002.

[76] S. Ma and J. Huang. Penalized feature selection and classification in bioinformatics. *Brief. Bioinformatics*, 9(5):392–403, 2008.

[77] J.M. Maris, M.D. Hogarty, R. Bagatell, and S.L. Cohn. Neuroblastoma. *Lancet*, 369:2106–2120, 2007.

[78] P. Massart, A. Barron, and L. Birge. Risk bounds for model selection via penalization. *Proba.Theory Relat.Fields*, 113:301–413, 1999.

[79] P. Mathe and S. Pereverzev. Moduli of continuity for operator monotone functions. *Numerical Functional Analysis and Optimization*, 23:623–631, 2002.

[80] K.K. Matthay, J.G. Villablanca, and et al. Seeger, R.C. Treatment of highrisk neuroblastoma with intensive chemotherapy, radiotherapy, autologous bone marrow transplantation, and 13-cis-retinoic acid. *Childrens Cancer Group. N Engl J Med*, 341:11651173, 1999.

[81] Charles A. Micchelli and Massimiliano Pontil. Learning the kernel function via regularization. *J. Mach. Learn. Res.*, 6:1099–1125, 2005.

[82] Charles A. Micchelli and Massimiliano Pontil. Feature space perspectives for learning the kernel. *Mach. Learn.*, 66(2-3):297–319, 2007.

[83] S. Mosci, L. Rosasco, and A. Verri. Dimensionality reduction and generalization. In *Proceeding of ICML 2007*, Corvallis, OR, June 2007.

[84] S. Mosci, M. Santoro, A. Verri, S. Villa, and L. Rosasco. A new algorithm to learn an optimal kernel based on fenchel duality. In *Proceedings of NIPS 2008, workshop on multiple kernel learning*, 2008.

[85] S. Mosci, M. Santoro, A. Verri, S. Villa, and L. Rosasco. Simple algorithms to solve sparsity based regularization via fenchel duality. In *Proceedings of NIPS 2008, workshop on optimization theory*, 2008.

[86] A. Navot, R. Gilad-Bachrach, Y. Navot, and N. Tishby. Is feature selection still necessary? In Craig Saunders, Marko Grobelnik, Steve R. Gunn, and John Shawe-Taylor, editors, *SLSFS*, volume 3940 of *Lecture Notes in Computer Science*, pages 127–138. Springer, 2005.

[87] G. Obozinski, B. Taskar, and M.I. Jordan. Multi-task feature selection. Technical report, Dept. of Statistics, UC Berkeley, June 2006.

[88] C. M. Perou, S. S. Jeffrey, M. van de Rijn, C. A. Rees, M. B. Eisen, D. T. Ross, A. Pergamenschikov, C. F. Williams, S. X. Zhu, J. C. Lee, D. Lashkari, D. Shalon, P. O. Brown, and D. Botstein. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proc Natl Acad Sci U S A*, 96(16):9212–9217, Aug March 1999.

[89] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. In *Science*, volume 247, 1990.

[90] S. Raudys. On dimensionality, sample size, and classification error of nonparametric linear classification algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):667–671, 1997.

[91] R. M. Rifkin and R. A. Lippert. Value regularization and fenchel duality. *Journal of Machine Learning Research*, 8:441–479, 2007.

[92] R. Rosipal, L.T. Trejo, and A. Cichocki. Kernel principal component regression with em approach to nonlinear principal components extraction. Technical report, CIS, University of Paisley, 2000.

[93] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[94] Robert E. Schapire and Yoram Singer. Improved boosting using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[95] B. Scholkopf, A. Smola, and K.R. Muller. Kernel principal component analysis. In *Advances in kernel methods - Support vector learning*, pages 327–352. MIT Press, 1999.

[96] M. R. Segal. Microarray gene expression data with linked survival phenotypes: diffuse large-b-cell lymphoma revisited. *Biostatistics*, 7:268–285, 2006.

[97] M. R. Segal, K. D. Dahlquist, and B. R. Conklin. Regression approaches for microarray data analysis. *J. Comput. Biol.*, 10:961–980, 2003.

[98] G.L. Semenza. Hif-1 and tumor progression: pathophysiology and therapeutics. *Trends in Molecular Medicine*, 8:6267, 2002.

[99] J. Shawe-Taylor, C. Williams, N. Cristianini, and J. Kandola. On the eigenspectrum of the gram matrix and the generalisation error of kernel pca. In *IEEE Transactions on Information Theory 51*, pages 2510–2512, 2004.

[100] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.

[101] M. Stone and R.J. Brooks. Continuum regression: Cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal components regression. *Journal of the Royal Statistical Society*, 52:237–269, 1990.

[102] Pablo Tamayo, Donna Slonim, Jill Mesirov, Qing Zhu, Sutisak Kitareewan, Ethan Dmitrovsky, Eric S. Lander, and Todd R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proceedings of the National Academy of Sciences, USA*, 96:2907–2912, 1999.

[103] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature. Genet.*, 22:281–285, 1999.

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

[105] C.J. Thiele. *Neuroblastoma*, pages 21–22. London, Great Britain: Kluwer Academic Publishers, 1999.

[106] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 56:267–288, 1996.

[107] P. Toronen, M. Kolehmainen, G. Wong, and E. Castren. Analysis of gene expression data using self-organizing maps. *FEBS Letters*, 451(2):142–146, May 1999.

[108] G.V. Trunk. A problem of dimensionality: A simple example. In *PAMI*, volume 1, 1979.

[109] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer Verlag, New York, 1995.

[110] J.P. Vert. *Kernel methods in genomics and computational biology*, pages 42–63. Camps-Valls, G., Rojo-Alvarez, J.-L. and Martinez-Ramon, M., 2007. Idea Group.

[111] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal on Computer Vision*, 57(2):137–154, 2004.

[112] J.L. Weinstein, H.M. Katzenstein, and S.L. Cohn. Advances in the diagnosisand treatment of neuroblastoma. *Oncologist*, 8:278–92, 2003.

[113] N. Weiss and M. Hassett. *Introductory Statistics.* Addison-Wesley, Reading, MA, 1982.

[114] J. Weston, A. Elisseeff, B. Schoelkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[115] J. Weston, S. Mukherjee, M. Chapelle, O. and. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. In *In Advances in Neural Information Processing Systems*, volume 13, 2001.

[116] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for l1-minimization with applications to compressed sensing. *SIAM J. Imaging Sciences*, 1(1):143–168, 2008.

[117] M. Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68(1):49–67, 2006.

[118] T. Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17:2077–2098, 2005.

[119] P. Zhao, G. Rocha, and B. Yu. Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, 2008. to appear.

[120] P. Zhao and B. Yu. On model selection consistency of lasso. *J. Mach. Learn. Res.*, 7:2541–2563, 2006.

[121] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.

[122] Z. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[123] L. Zwald, O. Bousquet, and G. Blanchard. Statistical properties of kernel principal component analyis. *Machine Learning*, 66:259–294, 2007.